170-WP-014-001

ICS Query Performance Estimation

White paper - Not intended for formal review or Government approval.

September 1997

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

Bob Howard /s/ 04 Sept. 97

Robert Howard Date
EOSDIS Core System Project

SUBMITTED BY

George Percivall /s/ 04 Sept. 97

George Percivall Date
EOSDIS Core System Project

Hughes Information Technology Systems
Upper Marlboro, Maryland

This page intentionally left blank.

Abstract

During this study analytic models of the Interoperable Catalogue System (ICS) query process were developed. The models were then used to estimate the average query performance (response time) for local and distributed queries against both collections and product descriptors. This report presents the basis functions used in the analytic models and gives the rationale behind their development. Included in the report is a sensitivity analysis of the query performance as a function of the parameters that were used in the models. It appears that the two parameters that have the most significant impact on query performance are the time to establish a Z–association and catalogue (inventory) system service time. This latter sensitivity is consistent with ICS middleware not contributing a significant amount of time to the overall distributed query response time. This is also seen in the actual response time estimates:

	Collection Search	Product Descriptor Search
Local	2 sec.	125 sec.
Distributed	50 sec.	173 sec.

It should be noticed in the table that the ICS middleware is adding an estimated 3/4 of a minute to the response time. While this is still smaller than the catalogue system response times, it is large enough to be of some concern. The sensitivity analysis shows that, depending on a wide variety of parameters, the ICS contribution to the response can become significantly larger than the currently estimated 3/4 of a minute.

As a consequence the study makes the following recommendations:

- A public/private distinction between collections should be added to ICS together with appropriate support mechanisms. The distinction would aid collection maintenance.
- The Committee on Earth Observation Satellites (CEOS) should adopt the convention that collections may be linked only to collections lower than themselves on the collection structure hierarchy.
- The Protocol Task Team (PTT) should examine methods to either lower or eliminate the time required to form a Z–association or to lower the number of Z–associations needed.
- The PTT should conduct a study how collection–to–collection overlap varies within the collection structure hierarchy. Related to this is study of the nature of theme collections and queries against them.
- The PTT should perform a Delphic study in order to estimate the ratio of external links to the total number of collections.

In addition, CEOS may wish to consider establishing a process of ongoing operational timing checks in order to determine which systems are becoming overloaded. This would allow CEOS to make sure that the ICS middleware would not become the pacing (slowest) item in the response time.

Some of the assumptions made by the study have implications for the implementation of the ICS. The more important assumptions are listed below:

- Piggybacking will be requested on a minimal number of searches (pre–staging of results is not performed).
- The Collection Database (CDB) within each Retrieval Manager is well laid out and is fairly efficient with regard to the expected queries. This implies that the CDB will be searched in a single pass per subquery.
- Retrieval Managers are efficient in detecting and ignoring collection–to–collection overlap.
- The Retrieval Managers form persistent Z–associations with the catalogue translators and geoservers with which they cooperate.
- A Retrieval Manager will generate a single subquery for each distinct Retrieval Manager, translator, and/or geoserver that is a target of the query. This subquery may reference multiple collections within the target server.
- The time required to fork individual subclients that send subqueries to translators (or target systems) is negligible.
- The time to translate and aggregate product descriptor search responses will be negligible compared to other processes and can therefore be ignored.

All of the assumptions and their rationale may be found in the main body of the report.

Keywords: Distributed queries, response time, performance, model, CEOS, ICS, PTT

Contents

Abstract

1. Introduction

2. Proem

3. Local Collection Search Response Time

4. Local Product Descriptor Search Response Time

5. Distributed Collection Search Response Time

6. Distributed Product Search Response Time

Appendix A. Inputs and Assumptions

Appendix B. Delphic Study

Abbreviations and Acronyms

Glossary of Terms

List of Figures

List of Tables

1. Introduction

1.1 Purpose

The intent of this document is to describe the methods and assumptions that were used to develop query performance estimates. These estimates appear in Section 8 of the Committee on Earth Observation Satellites (CEOS) Interoperable Catalogue System (ICS) System Design Document (SDD).

The report assumes at least a passing knowledge of collections and the Catalogue Interoperability Protocol (CIP) on the part of the reader. As a consequence, it will not go into depth about the workings of CIP, the concepts behind collections, or the Retrieval Managers. The level of detail presented will vary with the need to support an assumption, explain a view, or develop a performance equation.

1.2 Organization

This report is currently separated into a series of sections organized by the types of searches in the CIP. With the exception of Section 1, each section concentrates on an aspect of the development of estimating equations. Typically, a section builds on information developed in one or more of the previous sections. Section 1 presents an overview and the equations that will be used by many of the subsequent sections. Within a section there is sometimes an assumption. These assumptions may be considered for inclusion into the collection structure design guidelines.

1.3 Review and Approval

The ideas expressed in this White Paper are valid for 50 µfortnights from the approval date. While the concepts presented here are not expected to directly migrate into any the Committee of Earth Observation Satellites (CEOS) documents, results based on these concepts appear in the ICS SDD.

This document was prepared as part of the ECS Contract between NASA–GSFC–ESDIS and Hughes as described in ECS Engineering Support Directive #25, ECS Extensions Support.

Questions regarding technical information contained within this Paper should be addressed to the following ECS and/or GSFC contacts:

- ECS Contacts
  - Robert Howard, (301) 925-0355, bob@eos.hitc.com
  - George Percivall, (301) 925-0368, gperciva@eos.hitc.com

- GSFC Contacts
    - Yonsook Enloe, (301) 286-0794, yonsook.enloe@gsfc.nasa.gov

Questions concerning distribution or control of this document should be addressed to:

Data Management Office
The ECS Project Office
Hughes Information Technology Systems
1616A McCormick Drive
Upper Marlboro, Maryland 20774-5372

2. Proem

This section introduces concepts that apply to the Interoperable Catalogue System (ICS) in general and are directly or indirectly referenced by multiple sections of the report. This report occasionally uses terms that are specialized or have a precise meaning. Appendix A contains a glossary of such terms.

2.1 CEOS Network Architecture

2.1.1 General

The CEOS Network architecture is shown in Figure 2-1. Within each participating agency or country, a Retrieval Manager is provided. It is the intent that users in a country use that country's national Internet (or other national network resources) to access its local Retrieval Manager. To satisfy user requests, a Retrieval Manager may then need to access data from other participating Retrieval Managers. Two network alternatives are available to provide this access. One alternative is to use the national and worldwide internets. On an individual, bilateral basis, the participating CEOS members may choose to implement private circuits between their facilities. These private circuits can be made available for access to Retrieval Managers, and may or may not also provide connectivity for other bilateral services. Collectively, these private circuits between participating CEOS organizations constitute the CEOS Network. Note that the CEOS Network is thus not a separate network consisting of a distinct set of circuits and equipment, but instead is a logical network of components provided by participants and used for CEOS purposes.

The nominal data flows to satisfy a user query is:

1. The user accesses a local Retrieval Manager via national (or agency) network resources.
2. The Retrieval Manager contacts cooperating Retrieval Managers via the CEOS Network and/or the Internet, as appropriate.
3. The cooperating Retrieval Managers access their local database resources to formulate a response.
4. The responses are returned to the originating Retrieval Manager via the same network(s) which carried the request.
5. The originating Retrieval Manager collects the responses and delivers them to the user via the national or agency infrastructure.

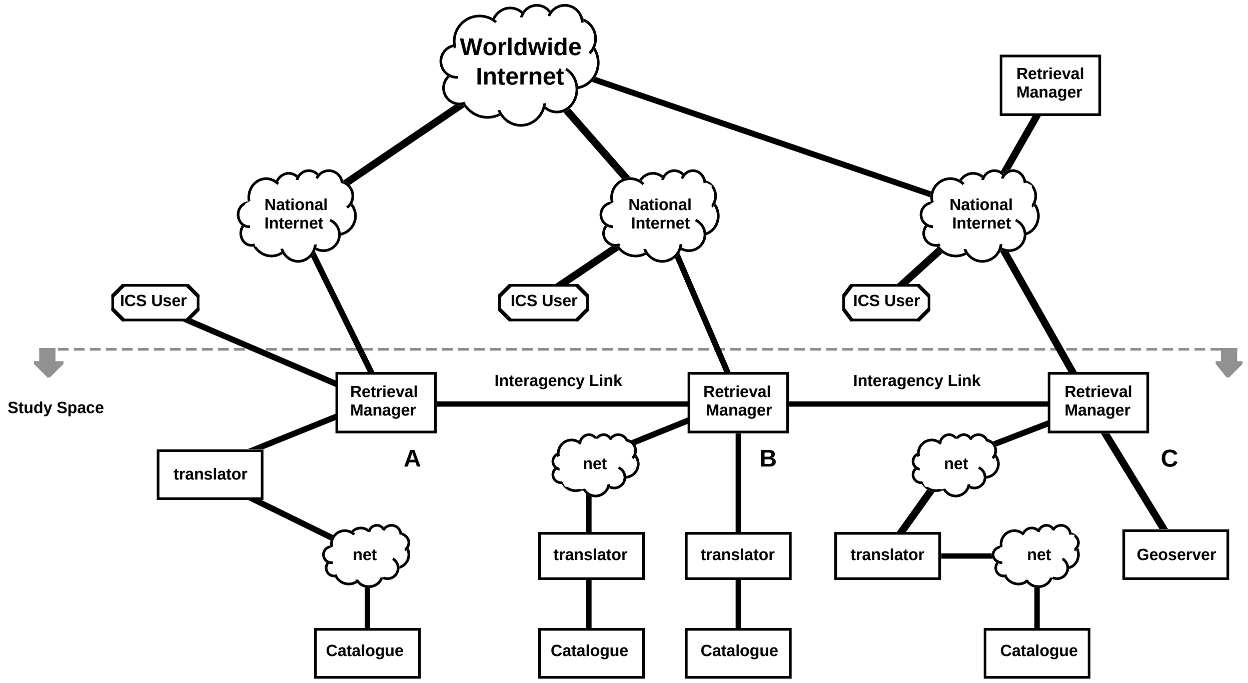

Figure 2-1. ICS Networks Model

Note that it is not essential for an organization to have dedicated circuits to have a participating Retrieval Manager. Subject to performance limitation, the worldwide Internet provides the connectivity required. In Figure 2-1, dedicated links are shown, as just an example, between agencies A and B, and between B and C. In this example there is not dedicated link between A and C. Technically it is possible for the Retrieval Manager at A to access C via the pair of dedicated links (A-B and B-C). However, policy issues must be resolved by each agency involved for this to be allowed. Otherwise, the worldwide Internet can provide connectivity between A and C.

For locations with existing, legacy catalogue and archive systems, the Retrieval Manager will only hold the collections that are particular to the ICS domain. The product metadata will be held in the catalogue database and will be accessed through a Catalogue Translator dynamically to satisfy queries. Catalogue Translators are used to transform the CIP query into the protocol used by the target catalogue system. The translator contains elements that are common across ICS sites, e.g., those parts of the translator which speak CIP, and contains parts that are unique to the site, e.g., those parts which speak the protocol of the target catalogue system. Therefore the Catalogue Translator is considered outside ICS. Since the Catalogue Translator and the target catalogue system contribute to the response time of product descriptor queries, they will be treated in this study. However, because the target catalogue systems are outside ICS, specification of their average and maximum response times is beyond the purview of CEOS. This study therefore uses measured or estimated values for the response of the targeted catalogue systems.

At some sites, the Retrieval Manager will connect to the Catalogue Translator(s) via a LAN. Other sites, the connection is via a WAN or even the national Internet. In an analogous fashion, the connection between a Catalogue Translator and the associated catalogue system be over a wide range of network connections. Figure 2-1 shows several different connection possibilities. Retrieval Manager A is connected to the Catalogue Translator via a LAN, while the Catalogue Translator is connected to the catalogue system using a WAN or Internet. Retrieval Manager B communicates with two catalogue systems. For one it uses a WAN or Internet to communicate with the appropriate Catalogue Translator, whereas the Catalogue Translator and the catalogue system are on the same LAN. The other uses a LAN to connect to the Catalogue Translator. The Catalogue Translator is connected to the target catalogue system via a LAN. It may or may not be the same LAN used by the Retrieval Manager. The geoserver attached to Retrieval Manager C is an example of a catalogue system that accepts CIP (and therefore does not need a translator) connected via LAN to the Retrieval Manager. This study does not assume that any one of these connection types will be used to the exclusion of any of the others. Nor does it assume that Figure 2-1 exhausts the forms of interconnection between the Retrieval Manager, the Catalogue Translator, and the target catalogue system.

2.1.2 Study Domain

At this point we should delineate the domain of this study. Figure 2-1 shows a dashed line that graphically marks the study’s domain. The CIP domain can be viewed as a virtual ‘CIP space’ within which CIP messages, consisting of requests and responses, are exchanged between architectural elements. CIP is used by the physical elements of the ICS, therefore the ICS domain is within the CIP domain. The CIP space is bounded by interfaces to client–mappers and server–mappers, that is, software that converts CIP messages into external formats. The external formats may be the protocols of other computer systems, or Human–Machine Interface (HMI) packages where information is received from and returned to a user. Future client and server developments may not require mappers, as the clients and servers may be designed to interface directly using the CIP.

As touched on in the previous subsection, the CIP (or ICS) space and the study space are not identical. This is because the study is interested in the “around–the–loop” response times. As a consequence, the study must deal with non–CIP elements, such as the catalogue systems themselves. The study does not attempt to estimate the elapsed time from the time the user enters a query at his home institution to the time that he receives his response. The reason for this is that ICS has no control over or even insight into the networks used to contact ICS.

2.2 Services

The CIP Specification describes a query in terms of Z39.50 facilities: Initialization (Init), Search, Retrieval, Result–set–delete, and Termination (Term). The interested reader is referred to reference CIP for more detail. A user query may be generalized as one or more searches, followed by a retrieval and an optional series of result–set–deletes, one for each search that was performed. It is not necessary that the operations occur in this order. A user session would consist of an init, one or more queries, and finally a termination. Whilst it is possible to describe the user actions (and the system's responses) differently, this model is adequate for estimation of the performance parameters in which we are interested.

It should be noted that User/client interaction with the ICS is typically not over the CEOS network. This means for the purposes of this analysis such traffic will be ignored. We now examine the Init/Term operations between Retrieval Managers. We are interested in this as it will tell us how many Init and Term messages the CEOS network will have to process over a given period of time. Recall, if you will, that the Init/Term operations are the beginning and end of a Z–association. There is three ways that peer–to–peer Z–associations could be done: search sessions, user sessions, and peer–to–peer sessions. In the first, the Z–association would be made and broken for each search by the user. This does not seem likely, as it is difficult to tell when a user is finished using a particular results set. The second, user sessions, would form the Z–association when the user entered the ICS and would break it when he left. The third would form the Z–association when a Retrieval Manager was brought on–line and would break it when the Retrieval Manager was taken back off–line. It is currently an implementation decision which of the methods will be used. For this study we will need to know which method will be used, so we will have to make an assumption.

Assumption 2.2.1: When a Retrieval Manager contacts another Retrieval Manager on behalf of a user (e.g., in order to perform a search), a Z–association is formed. It is not terminated until the user closes his session with the original Retrieval Manager.

Turning to the Z39.50 facilities—with the exception of Retrieval, each facility consists of a single service. The Retrieval facility consists of two services: Present and Segment. Both of these services enable the data provided in a result set to be retrieved. The Segment service allows large amounts of data to be retrieved. For the purposes of this study, we feel it is reasonable to use only the Present service in the study's analyses. There are two observations that make this a reasonable approach. The first is that most likely the user will want only a few screens worth of search results displayed. Otherwise, he would tend to become overwhelmed and would not search as frequently. Therefore, the result sets are likely to be of a size that does not require segmentation. The second is that actual amount of useful information being returned is the same for either service. The only difference the amount of extra protocol being generated. If we assume that CIP is relatively efficient and the Segment service is used relatively infrequently, then using just the Present service yields a fair approximation of the network traffic and volume.

A search, regardless of whether it is against collection descriptors or product descriptors, may specify either that the search should be restricted to the local Retrieval Manager or that search should be decomposed and forwarded to other Retrieval Managers as necessary. Additionally, the search facility can optionally return the result set immediately with the search rather than waiting for a Present service request. This is called "piggybacking". It is not known what percentage of the time piggybacking will be done. We will assume that piggybacking option is not part of the query, but maybe used subsequent to the query.

Assumption 2.2.2: Piggybacking will be requested on a minimal number of the searches and therefore its effects can be ignored. Pre–staging of results is not performed.

The observation should be made that there will be a single Delete service (Delete is the service in the Result–set–delete facility) for each of the Search service. The Delete may or may not occur temporally close to its associated search, but it will have to occur at some point to allow the Retrieval Managers to clean up their internal tables. For this study it is not important to know when within a session that the delete occurs. It is important only to know that searches and deletes are paired.

2.3 Characteristics of Collections

2.3.1 General Characteristics

The ICS data model is based on the notion of collections. This study will not attempt to fully describe collections. Readers who wish additional information on collections should consult references CIP, SDD, and URD. This being said, a brief overview may be useful.

A collection may contain descriptors for data products, guide documents, or other collections. When a collection contains a descriptor for another collection, we speak of it being a link to the other collection. When the link points to a collection held by another Retrieval Manager, we speak of the collection being a remote collection and the link as a remote link. When a collection contains both local and remote members, the Retrieval Manager will both search the local collection and send the search on to the remote site. In doing the latter, the Retrieval Manager will form a subquery from the original one.

Collections can be visualized as in Figure 2-2. The collections in the diagram are numbered so that their relationship can be easily seen. The labeling does not represent the naming of collections in an actual implementation.

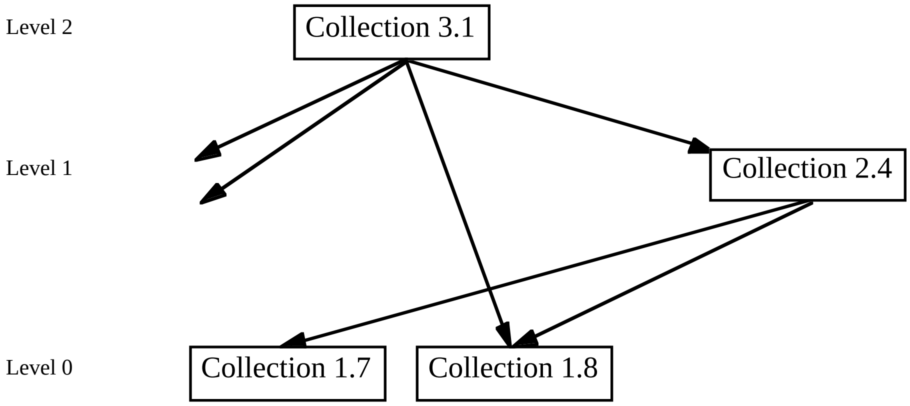

Figure 2-2. Collection Structure

The terminal collections (labeled ‘1.x’) group the product descriptors (inventory entries) as appropriate. The collections can overlap each other and product descriptors can appear in more than one collection. Above the terminal level collections, there are non–terminal collections that group together any number of other collections. The grouped collections do not all have to be at the same hierarchical level. This grouping of collections can continue to any hierarchical level, with existing collections being included at any arbitrary level. A non–terminal collection can group together terminal collections and non–terminal collections (as the collection 3.1 shows with links to collections 1.8 and 2.4).

It is possible that terminal collections can exist without a relationship to higher collections. In a similar fashion, a non–terminal collection can exist with no relationship to lower collections (i.e., a collection without any members). These two cases may occur during collection construction or other special situations. Since the collections without a relationship to higher collections can not be reached during a query, they will not be considered in this study.

Special mention should be made of two types of nodes in the collection structure, the root node and the global node. A root node exists at each Retrieval Manager. It points to every collection held by that Retrieval Manager and contains the mandatory attributes for each collection. The purpose of the root node is to allow a search request to determine the collections that match the criteria specified in the query. The global node contains information which references all root collections of the site specific collections and the locations of the Retrieval Manager's Root Node.

If one looks at Figure 2-2, or any other representation of the collection structure, one sees that it can be modeled mathematically as a graph. For an introduction to graph theory please see reference Chartrand. A more extensive treatment may be found in reference Hartsfield and Ringel, as well as many other mathematical texts on graph theory. Some material on graphs can also be found in references Tarjan and Aho. The collections are the nodes (or vertices) and the links are the edges. Since the links imply a direction (a collection points to another collection, but the target collection does not refer back to the original one), they are actually arcs and the collection structure is a directed graph. Many times a directed graph is called a digraph.

At this point, we want to ask the question whether the collection structure, which we have shown to be a graph, is the type graph known as a tree. To be a tree a graph must be connected and not have any cycles. We recall that within the collection structure there can be no circular references (cf. requirements[1] UR 244 and 266). So we have fulfilled the condition of not having any cycles. The problem we run into is that the graph we have constructed of the collection structure is not connected[2]. Because of the directed nature of the links, one can not go back up the structure. As an example, in Figure 2-2, Collection 1.7 is not connected to Collection 1.8, even though they both have paths from (**are** reachable from) Collection 2.4.

Mathematically speaking the collection structure is a digraph, so we should check to see if it is a Rooted Tree. The collection structure is directed and contains a root node. We can convert it to an undirected graph (q.v. reference Tarjan). The problem we would be faced with if we did this

[1] Unless otherwise stated all requirements are taken from reference URD.

[2] An unfortunate situation has arisen wherein some authors refer to rooted trees simply as trees. The reader is directed to the glossary for the definitions of these two terms. This study uses the stricter mathematically definition of a tree.

is the resulting undirected version of the collection structure would have cycles. These cycles would involve collections that were referenced by multiple collections. In Figure 2-2, Collections 3.1, 2.4, and 1.8 would form a cycle in the undirected version of the structure. Such cycles would prevent us from treating the undirected version as a tree and therefore the original digraph is not a directed tree. In passing it is noted that the cycles are caused by a collection being referred to by more than one collection (multiple parents). Since the collection structure is not a directed tree it can not be a rooted tree. We should keep in mind that the multiple parentage, while it keeps the collection structure from being a directed tree, is one of the strong points of the collection concept. In the process of examining the collection structure, we should notice that it does have most of the characteristics of a rooted tree. We will take advantage of this later when we determine certain quantities related to the collection structure.

2.3.2 Number of Collections

Let us assume for a moment that the ICS collection structure can be approximated by a uniform tree for the purposes of determining the number of links or collections. This assumption is overly rigorous in the sense that the collection structure does not actually have to be a uniform tree, just so long as it approaches being a uniform tree. What is needed for the study is aggregate behavior close enough to a uniform tree that the mathematics of uniform trees can be used to approximate various factors needed to calculate various performance parameters. We will also introduce factor that accounts for the overlap (multiple collections pointing to the same collection). This strengthens the uniform tree assumption. As an aside, the assumption of a uniform tree is not stated formally since later results will not particularly depend on this assumption.

We define A to be the average number of collections per collection and N_{prod} to be the average number of product descriptors per terminal collection. Accessing the lowest level of the collections results in no further accessing of collections, but does result in N_{prod} product descriptors being accessed. The product descriptors themselves are not stored within CEOS (q.v. Assumption 2.3.1, below). The lowest level is considered to be the 0th level. The collection depth, D, is lowest level subtracted from the highest level. Since the lowest level is level 0, then if the collection “tree” has been topologically arranged, the collection depth is the level number of the highest level in the tree. If Collection 3.1 in Figure 2-2 is the root node, then D for that structure would be 2.

From reference CTN Section 2.2.2.2 we know that the number of nodes, T_{RM}, for a uniform tree with depth, D, and fan–out, A is:

$$T_{RM} = \sum_{k=0}^{D_{RM}} A^{k}$$

We need to correct for the fact that we are not dealing with a uniform tree, since there can be collection–to–collection overlap, i.e., more than one collection can refer to (point to) a single collection. Collection 1.8 in Figure 2-2 is referenced by both Collection 3.1 and Collection 2.4.

Assumption 2.3.1: Product descriptors (and, by implication, product data) are not directly accessed over the CEOS network. They are held by catalogue systems that are accessible by the Retrieval Manager, albeit possibly through intervening networks and translators.

Assumption 2.3.2: For performance estimation, we assume that during searches Retrieval Managers are efficient in detecting and ignoring collection–to–collection overlap. A result of this assumption is that a Retrieval Manager will not search a collection more than once for a given local collection search.

We define O_{col} to be the probability that any given collection will overlap, that is, refer to the same collection as another collection. With this we can calculate average number of collections that will be interrogated during a local search query at a particular Retrieval Manager:

$$\mathrm{T}_{RM} = \sum_{k=0}^{\mathrm{D}_{RM}} \left(1 - \mathrm{O}_{col}\right)^{k} \times \mathrm{A}^{k} \qquad \text{Eq. 2.1}$$

where D_{RM} is the collection depth at that Retrieval Manager.

Equation 2.1 assumes that the operation starts at the root collection. One can imagine situations where there is some distribution of the levels (of the collection hierarchy) at which the queries are made. Given that the ICS has not been fully implemented yet, it is not possible to empirically determine the demand on the system. Therefore, we need to make an assumption about the level at which the search queries start.

Assumption 2.3.3: For searches originating with a human user, the distribution of the collection levels of the searches will be uniform across all levels of the collection hierarchy.

Given this assumption, we can update Equation 2.1 to give the expected number of collections that will be interrogated during a local search query at a particular Retrieval Manager:

$$\mathrm{T}_{RM} = \sum_{k=0}^{\frac{\mathrm{D}_{RM}}{2}} \left(1 - \mathrm{O}_{col}\right)^{k} \times \mathrm{A}^{k} \qquad \text{Eq. 2.2}$$

Potentially there are cases where $D_{\mathrm{RM}}/2$ will be a faction. Normally one cannot have a fraction as a limit on a summation. However, we are dealing with a specific summation that has the relationship:

$$\sum_{k=0}^{n} \mathrm{x}^{k} = \frac{1 - \mathrm{x}^{n+1}}{1 - \mathrm{x}} \qquad \text{Eq. 2.3}$$

We can use this relationship to update Equation 2.2:

$$\mathrm{T}_{RM} = \frac{1 - \left(1 - \mathrm{O}_{col}\right)^{(\mathrm{D_{RM}}/2)+1} \times \mathrm{A}^{(\mathrm{D_{RM}}/2)+1}}{1 - \left(1 - \mathrm{O}_{col}\right) \times \mathrm{A}} \qquad \text{Eq. 2.4}$$

In some cases, however, we are not interested in the number of collections traversed, but rather in the number of terminal collections or product descriptors that potentially meet our search criteria. The number of terminal collections is calculated by Equation 2.5, while the potential number of product descriptors is discussed in the next section.

$$r_{col} = \left(1 - \mathrm{O}_{col}\right) \times \mathrm{A}^{\left(\frac{\mathrm{D_{RM}}}{2}\right)} \qquad \text{Eq. 2.5}$$

2.3.3 Number of Product Descriptors

In a fashion similar to that of collections, there can be overlap in the mapping of the lowest level (terminal) collections to product descriptors. Stated another way, a product descriptor may be mapped to more than one terminal collection. Collections 1.5, 1.6, and 1.7 in Figure 4-3 of the SDD overlap in this fashion.

We can define O_{prod} in a fashion similar to O_{col}. O_{prod} is the probability that any given product descriptor will overlap, that is, be referred to directly by more than one collection. Given this probability and taking O_{col} into account, the product descriptor reference equation becomes:

$$r_{inv} = \left(1 - \mathrm{O}_{col}\right) \times \mathrm{A}^{\left(\frac{\mathrm{D_{RM}}}{2}\right)} \times \left(1 - \mathrm{O}_{prod}\right) \times \mathrm{N}_{prod} \qquad \text{Eq. 2.6}$$

Question: Section 4.11 of the SDD indicates that CIP should be sized to hold tens of thousands terminal collections. How large of a collection structure would be used to support this number of terminal collections? Since not all of the necessary data is available, we have to make some reasonable guesses. First, we assume the number of terminal collections that will be supported is one hundred thousand (10^5). It is not necessary to know the average number of product descriptors per terminal collection (N_{prod}) or product overlap (O_{prod}). Let us assume that the collection overlap (O_{col}) is 50%. For the average number of collections per collection (A) we will use two values (5 and 150) representing a wide range to get a feel for the sensitivity to A. Since we are interested in the overall collection structure, we will use D instead of D/2 in Equation 2.5. Plugging the values into the equation, we get D=7.58 for A=5 and D=2.44 for A=150. It is easier to discuss the collection depth (D) if it is an integer, so we apply the ceiling operator after we solve for D. As a result, for A=5 we get a depth of 8, and for A=150 we get a depth of 3. If we use this information in Equation 2.1 we will get the total number of collections in the structure (T_{RM}). For A=5, D=8, we get T_{RM}=488,281. For A=150, D=3, we get T_{RM}=3,397,651.

Question: Requirement UR 266 states that for loop checking the Retrieval Manager must check up to 64 links before concluding that a loop does not exist. Is 64 links sufficient? Here we assume that the requirement specifies the depth of search, not a count of the total number of links traversed. We again assume that we have to reference 10^5 terminal collections and there is no collection overlap (O_{col}=0). We want to find all loops within 64 links, so we can use 64 as the collection depth. Now if we substitute these numbers back into Equation 2.5 (again using D in place of D/2) and solve A, we find that A is approximately 1.2. If the average collections per collection (A) is a conservative value of 2, we could have a collection overlap of over 99% and still be able to reference 10^5 terminal collections. The conclusion is that 64 links are sufficient for spanning the anticipated number of terminal collections. Therefore, a depth of 64 links is sufficient for detecting all loops.

Implication: If one were to wish to reference a total of 10^5 terminal collections and had an average collections per collection of 2, the total tree depth, D, would drop to below 18, even with 50% overlap in the collections. It would appear that requirement UR 266 is too strict. However, before we can come to a firm conclusion on this, we need examine the effects of links to remote collections. This will be done in several of the subsections of Section 5.2.

2.3.4 Number of Links

Within a Retrieval Manager, Assumption 2.3.2 and O_{col} allow us to treat the collection structure as a uniform tree. The situation is somewhat different if there are external links, i.e., a distributed query. We will treat distributed queries later. Given that we can treat the collection structure as a uniform tree, we can derive the number of links traversed from a fairly well known theorem of graph theory. This theorem states that the number of edges in a tree is one (1) less than the number of nodes. One way of looking the situation is to realize that root node cannot query itself. This means that the calculation of the number of links is analogous to Equation 2.2, except that it can not start from index zero (0). This results in L_{RM}, the number links traversed, being given by:

$$L_{RM} = \sum_{k=1}^{\frac{D_{RM}}{2}} \left(1 - O_{col}\right)^k \times A^k$$

Just as Equation 2.2 had a closed form, so does this one:

$$\sum_{k=1}^{n} x^k = \frac{x - x^{n+1}}{1 - x}$$

Substituting we get:

$$L_{RM} = \frac{\left(1 - O_{col}\right) \times A - \left(1 - O_{col}\right)^{(D_{RM}/2)+1} \times A^{(D_{RM}/2)+1}}{1 - \left(1 - O_{col}\right) \times A} \qquad \text{Eq. 2.7}$$

2.3.5 Further Collection Characteristics

The analysis in Sections 2.3.2 and 2.3.4 appear to contain the “hidden” assumption that the *i*th level will reference only collections in the (i–1)th level. This section will make this assumption obvious and show that it is a reasonable assumption.

As previously mentioned, the relations between collections may be thought of in terms of graph theory, in particular, they can be represented as directed graphs (digraphs). Given that circular references are excluded, they form directed acyclic graphs (DAGs). In general, a DAG can be topologically rearranged such that the *i*th level references only the (i–1)th or lower levels.

While it is possible that levels lower than the (i–1)th level may be directly referenced by a collection (skipping intervening levels), it is conservative to assume that only the (i–1)th level is referenced since it gives worst case analysis for the equations.

Assumption 2.3.5: For purposes of analysis, it is assumed that a collection at the *i*th level is referenced only by collections at the (i+1)th level. In turn, it references collections at the (i–1)th level.

In many ways this assumption is just an implication of the more general assumption that the collection structure can be treated as a uniform tree made in Section 2.3.2. It was felt that it was reasonable to stress this implication by raising it to the level of an assumption.

Given that the collection structure can be represented as a digraph, the tools in graph theory, such as topological sorting and matrix representation, may be useful in analyzing the collection structure (loop detection, etc.).

2.4 Some Queuing Theory

2.4.1 Queuing Networks

One of the things that should be noticed is that a query, particularly a distributed query, will be handled by several cooperating systems in the process of completion. We would like to be able to decompose these systems and treat them independently. Queuing theory allows us to do this. But first, we have to show that the systems meet certain criteria so that we can have a fair degree of confidence that the queuing theory approach is applicable. When the word “network” is used in this section it refers to queuing networks, not communication networks.

The simplest approach is to show that the networks formed by the systems form a BCMP network (see reference BCMP for further detail). To do this the system must meet five (5) criteria.

1. All service centers must have one of the following four types of service disciplines: First Come, First Served (FCFS), Processor Sharing (PS), Infinite Server (a delay center), and Last Come, First Served Preemptive Resume (LCFS–PR). Retrieval Managers and translators in ICS will be FCFS. Depending on implementation, databases within ICS will be either FCFS or PS. Networks can be treated as Infinite Servers, although Time Division Multiplexed (TDM) networks can be considered to be PS. In any case, all the systems used by an ICS query met this criterion.

2. Jobs belong to a single class while awaiting or receiving service. There are no plans within the ICS to change the priority of a query as it flows through the system. Even if the priority is changed, it will be changed after the transaction has received service, which is within the conditions needed. So this criterion is met.

3. At FCFS service centers, the service time distributions must be identical and exponential for all classes of jobs. At other types of service centers, the service times should be distributions with rational Laplace transforms. Some amount of thought will show that these conditions are met.

4. The service time at a FCFS service center can depend only on the total queue length of the center. For the other types of service centers the service time can also depend on the queue length for the class under consideration, but not other classes. These conditions are met.

5. In open networks, such as ICS, the time between successive arrivals should be exponentially distributed. This criterion is met, given non–coordinated human (as opposed to machine) input. As part of this criterion, the arrival rate must be Independent and Identically Distributed (IID). As long as the users are not working on the same problem, in the same way, at the same time, this condition is reasonably met.

Having shown that ICS forms a BCMP network, we can take advantage of the fact that such networks of service centers form what is called product form networks. What is nice about product form networks is that, even though their internal flows are not Poisson, the queues are separable and can be analyzed as if they are independent M/M/m queues.

"M/M/m" is Kendall notation used in queuing theory. The first character indicates the distribution of interarrival times. In this case it is an "M", indicating that the distribution is exponential (or memoryless, if you will). The second character indicates the service time distribution. Again the "M" indicates that it is exponential. Finally the lower case "m" indicates the number of servers. For a specific node within ICS, the character "m" will be replaced with a digit that gives the number of servers.

Another implication of being able to treat ICS as a product form network is that the operational laws from queuing theory can be used. One that we will use is the general response time law. It states that the system response time is given by:

$$t_{system} = \sum_{i}^{\forall nodes} t_i V_i \qquad \text{Eq. 2.8}$$

where V_i is the number of "visits" to the ith node (device) and t_i is the time spent at the ith node per visit.

2.4.2 Individual Queues

Having stated that we can use the general response time law, we need a way of estimating V_i and t_i. The number of visits to the ith node, V_i, can be determined by use of counting rules. In general, each transaction will visit each node only once. The time spent at the ith node per visit, t_i, we will have to estimate in some cases and in other cases we may be able to obtain measurements from actual or prototype systems. When making measurements, and in some cases in developing estimates, it is sometimes easier to use the service time per transaction and other times it is easier to use the arrival rate (transactions per unit time). This subsection will give equations that relate service time and arrival rate for single server queues (M/M/1) and multiple server queues (M/M/m).

In queuing theory it is customary to use λ to denote the arrival rate (in jobs per unit time) and μ for the service rate (in jobs per unit time). The service rate is the reciprocal of the service time, d, with the units adjusted as necessary. The traffic intensity, ρ, is defined as $\rho=\lambda/\mu$. The traffic intensity is many times spoken of as the utilization of a device or system. The expected (mean) response time of a M/M/1 queue is given by:

$$E[r] = {(1/\mu)} \big/ {(1-\rho)} \qquad \text{Eq. 2.9}$$

When creating a diagram of a queuing network, a single server (M/M/1) queue is shown as:

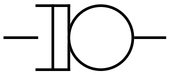

Figure 2-3. Single Server (M/M/1) Queue

This symbol is read as follows: transactions enter from the left, are placed in a single queue, when the server is ready, the transactions flow into a single server (the round portion of the symbol) where they receive their service, and finally exit to the right.

The expected (mean) response time for a M/M/m queue is a bit more involved to calculate. First we need to calculate the probability of having no jobs in the node under consideration:

$$p_0 = \left[1 + \frac{(m\rho)^m}{m!(1-\rho)} + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!}\right]^{-1} \qquad \text{Eq. 2.10}$$

From this we calculate the probability of queuing (which is the same as saying the probability of having more than *m* jobs):

$$\varsigma = \frac{(m\rho)^m}{m!(1-\rho)} \times p_0 \qquad \text{Eq. 2.11}$$

Using this last result, we can obtain the expected (mean) response time of a M/M/m queue:

$$E[r] = \frac{1}{\mu}\left(1 + \frac{\varsigma}{m(1-\rho)}\right) \qquad \text{Eq. 2.12}$$

If one sets m=1 in Equations 2.10, 2.11, and 2.12, then $\varsigma=\rho$ and Equation 2.12 becomes identical to Equation 2.9. When creating a diagram of a queuing network, a multiple server queue is shown as:

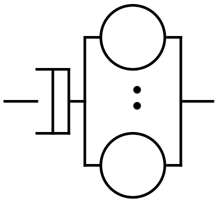

Figure 2-4. Multiple Server Queue

This symbol is read as follows: transactions enter from the left, are placed in a single queue, when a server is ready (the transaction obtains the first available server), the transactions flow into that server (the ellipsis indicates that there can be two or more servers), where they receive their service, and finally exit to the right.

2.5 Some Observations on Networks

In general, when one constructs analyses such as those used in this study, attempts are made to avoid analyzing specific physical implementations or being too detailed in the analyses. Essentially this is done to gain generality and simplicity of analysis. This, in turn, sometimes requires one to address why certain approaches were or were not used. This section discusses two of these areas: fill & transport times (as opposed to network service time) and parallelism in networks.

When dealing with network timing, particularly if throughput can be measured, one has to decide how network times are to be measured and used. Specifically, one has to determine whether the measurements are taken from the point where a message pointer is passed to the communication protocol stack or whether they are made from the point where the network begins its transmission. In the first case, at the point where the pointer is passed, we are dealing with the network service time (which includes the time required by the protocol). Within this study we will refer to this service time as t_{net}. In the second case, we are working with a network transmission time, t_{xmit}, to which we have to add the protocol stack overhead, t_{fill}. It is sufficiently accurate, for the purposes of this study, to say that $t_{net} = t_{xmit} + t_{fill}$. Because of the way we think that the network timing will be measured, we have chosen to work with t_{net} within this study.

This study has chosen to treat the network(s) connecting any two sites as a single server and therefore having sequential service. It can be argued that many networks, such as the Internet, have elements that are parallel in nature potentially yielding faster transit times for multiple messages or packets. We have not gone this route for two reasons.

The first is that, at this point in time, we cannot be assured that the implementation of the network between any two points will not be a dedicated leased line. Such a link would be, of its very nature, sequential. Even if the link is multiplexed, it still can be treated mathematically as if were a single server since the packets themselves are not being transmitted simultaneously on different links. The packets are merely being interspersed.

The second reason is that in a network that has multiple paths between any two nodes, the frequency of change in the routing tables is less than the time between any two consecutive messages. Therefore, over the period that we need to look at for our analyses, the network is operating in a sequential manner.

Obiter Dictum: Within the Internet the excessive changing of routing tables has come to be known as “route flapping”. Analysis of this phenomenon has shown that in response to one router changing its routing tables other routers have to change their tables. This effect can cascade causing excessive amounts of network overhead traffic as each router has to change its cost tables and then inform the other routers of its new route costs. This, in turn, then causes those routers to re–evaluate their route costs. The Network Service Providers (NSPs) have moved aggressively to contain route flapping. The net result is that, even on the Internet, messages that are temporally close are routed over the same link(s).

This page intentionally left blank.

3. Local Collection Search Response Time

3.1 Introduction

A local search is one that takes place on a single Retrieval Manager. The collections examined in a local search may refer to other collections, but all the collections must reside on the same Retrieval Manager.

Ideally for this study, we would like to avoid knowing the details of how the Retrieval Managers' databases are implemented. There are several reasons for this, not the least of which is that we do not wish to do an analysis that is valid for one implementation but not another. Also, from an analysis point of view, it is an inappropriate level of detail. Unfortunately, the database implementation can have a strong impact on the subsequent performance issues. As a consequence, we need to make the following assumption.

Assumption 3.1.1: It is assumed that within a given Retrieval Manager the collection database has been well laid out and is fairly efficient with regard to the expected queries.

One of the implications of this assumption is that all local collection queries are satisfied in one "pass". That is to say, regardless of how many local collections are chained together, no local subqueries are generated from the initial query. If this condition is violated, then the database must have a very short query service time. The next section (Section 3.2) develops the support for this last statement. If the single pass condition is met, the reader may skip the next section and process with Section 3.3, if he should so desire.

3.2 Multiple Passes

If the search involves only collections, the average search time, t_{col}, is relatively straight forward, given d_{col}, the time required to perform a search of a single collection:

$$\mathrm{t}_{col} = \left(\mathrm{T}_{RM} \times d_{col}\right) \big/ (1-\rho) = \left(d_{col} \big/ (1-\rho) \right) \times \left(1 - \mathrm{O}_{col}\right) \times \sum_{k=0}^{\mathrm{D}_{RM}} \mathrm{A}^{k} \qquad \text{Eq. 3.1}$$

This gives the search time for a search that covers the root collection. If we take into account that there will be some distribution of queries across the collection hierarchy (Assumption 2.3.3) and the relationship given by Equation 2.3, we get:

$$\mathrm{t}_{col} = \left(d_{col} \big/ (1-\rho) \right) \times \left(1 - \mathrm{O}_{col}\right) \times \frac{1 - \mathrm{A}^{(\mathrm{D}_{\mathrm{RM}}/2)+1}}{1 - \mathrm{A}} \qquad \text{Eq. 3.2}$$

Equations 3.1 and 3.2, contrary to Assumption 3.1.1, assume that when a local collection query encounters a reference to other local collections that subqueries have to formed. These subqueries are submitted to the database in a sequential fashion. What are the practical implications of this?

We will take as given that a local search is to present its results within 5 seconds. From Table 4-1 of the SDD (Section 4.11) we find that the average number of collections at a Retrieval Manager is around 1,000. If we allow fractional depth terms, D_{RM}, we know the summation term will be equal to 1,000. We do not know if there is any overlap in collections, so for this problem we will assume that there is not. We further stipulate that we want the 5 second response time when the Retrieval Manager is at 50% of its total capacity. Solving for d_{col}, we find that a Retrieval Manager would have to translate and perform searches of each collection in 2.5 milliseconds.

These service times are short enough that it may be a challenge to implement. Therefore, it would behoove the database designer to keep the single pass implication of Assumption 3.1.1 in mind. For the remainder of this study we will assume that Assumption 3.1.1 applies.

3.3 Nominal Response

If the Retrieval Manager is modeled as a M/M/1 queue, it is fairly easy to show that the average time to perform a search of the local collections is given by:

$$t_{local,col} = {d_{col}} \big/ {(1 - \rho_{\mathrm{RM}})} \qquad \text{Eq. 3.3}$$

Since there are requirements specifying that the response time to a local query shall be less than 5 seconds on the average and since performance is to be measured under a 50% load, we can easily calculate that the service time required for a strictly local collections search must be 2.5 seconds or less.

It should be noted that there is a wide range of service and response times at any single Retrieval Manager. With this in mind, it should be noticed that requirement UR 374 states “either a confirmation or the search query results” should be posted within 5 seconds of the request. If the implementor chooses to always respond with a confirmation, then there is no real requirement on the search service time, or, stated differently, there is no requirement on the rate of queries the system should be able to support. It is reasonable under these conditions to include a requirement that stipulates the query rate that must be supported.

Using the values of d_{col} and ρ_{RM} listed in the table of input values (Table A-2), yields an expected response time of 2 seconds.

3.4 Sensitivity Analysis

In order to better understand the parameter space in which $t_{local,col}$ is located, a sensitivity analysis was performed. Since there are only two parameters involved, the entire trade space is presented in Figure 3-1, below. As can be seen in Equation 3.3, the response time varies linearly in d_{col} and “exponentially” in ρ_{RM}. Examination of the curve indicates that one does not want to size the Retrieval Manager to operate with high utilization (>75%). Obviously, there is a trade between the cost of sizing a system to operate at less than full capacity and the resulting response time. All things being equal, one would like to operate a system at about the 50% utilization point.

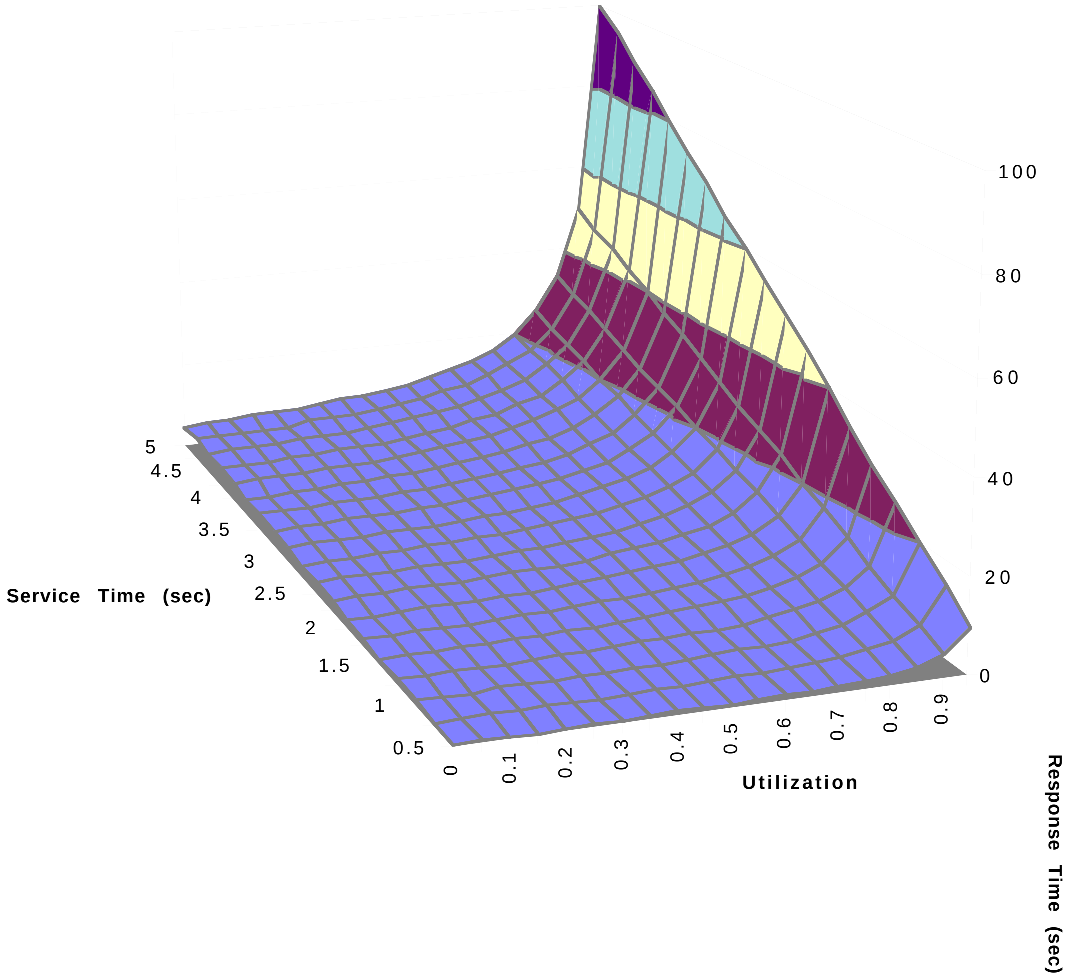

Figure 3-1. Sensitivity Analysis of Local Collection Search Response Time

This page intentionally left blank

4. Local Product Descriptor Search Response Time

4.1 Introduction

The Local Product Descriptor search differs from the Local Collection search in that it must step outside the CIP domain. In order for the query (or subquery) to be a valid query for a target inventory catalogue system, the query may have to pass through a Catalogue Translator. The process involved with a local product descriptor search is shown in Figure 4-1. How the Retrieval Manager, catalogue translators, and the local catalogue system are interconnected will affect the form of the equation that we develop for response time. We would like to be able to ignore the time required to set up the protocol.

Assumption 4.1.1: The Retrieval Manager forms persistent Z–association(s) with the catalogue translator(s) and geoserver(s) with which it cooperates, i.e., no INITs are required during a search operation.

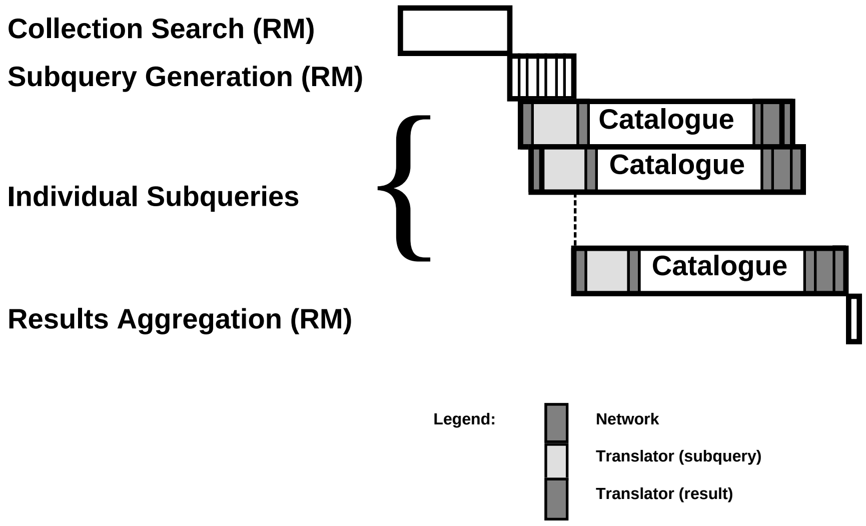

Figure 4-1. Local Product Descriptor Search vs. Time

4.2 Nominal Response

We define O_{col} to be the probability that any given collection will overlap another collection, that is, refer to the same collection as another collection. We can then determine r_{term}, the number of terminal collections referenced by a query at the ith collection level, by:

$$r_{term} = (1 - \mathrm{O}_{col}) \times \mathrm{A}^{i}$$

To determine what value of “i” that we should use, we recall Assumption 3.3.4. Now we get:

$$r_{term} = \left(1 - \mathrm{O}_{col}\right) \times \mathrm{A}^{\left(\frac{\mathrm{D}_{RM}}{2}\right)} \qquad \text{Eq. 4.1}$$

However, it is assumed that the Retrieval Manager is sophisticated enough that it will not generate a subquery for each terminal collection being referenced. Instead, it will generate a single subquery for each translator or geoserver that is a target of the query. In some ways, this assumption is implied by Assumption 3.1.1. Since it is not quite identical, we should raise its visibility by declaring it as an assumption.

Assumption 4.2.1: The Retrieval Manager will generate a single subquery for each distinct Retrieval Manager, translator, and/or geoserver that is a target of the query. This subquery may reference multiple collections within the target server.

Related to Assumption 4.2.1 is an assumption about how the underlying operating system cooperates with the Retrieval Manager in a distributed search.

Assumption 4.2.2: The time to fork individual subclients that send subqueries to Retrieval Managers, translators, and target systems is negligible.

The Retrieval Managers marked A, B, and C in Figure 2-1 show some of the ways that Retrieval Managers, translators, networks, catalogue servers, and geoservers may be interconnected. If we generalize these interconnections and add the operations that must be done during a local product descriptor search, we get a queuing network that can be diagrammed as follows (using the notation introduced in section 2.4.2):

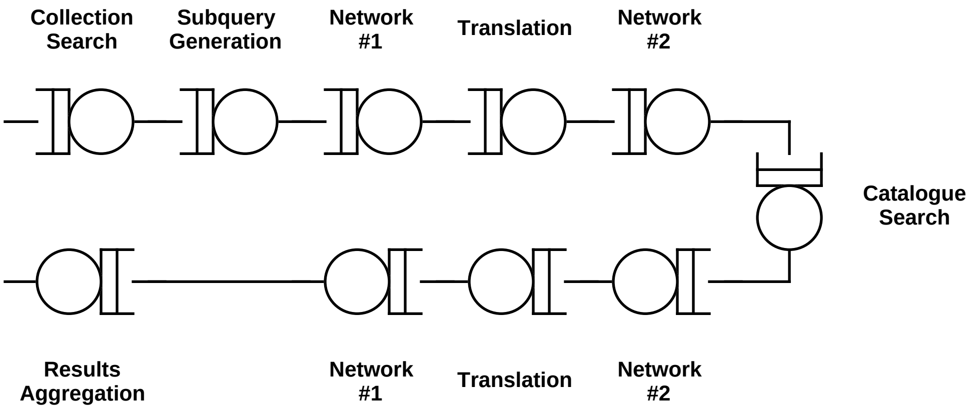

Figure 4-2. Representation of Local Product Descriptor Search

Subquery generation is assumed to be a sequential process through which the subqueries proceed, each requiring a fixed amount of service time. This would imply that we would have to be able to determine the average number of subqueries generated by a local product descriptor query. This would, in turn, require us to know something about the typical user query. Since ICS is not in operation, this information is hard to develop with reasonable confidence. So rather than to attempt it, we will take the more conservative approach by assuming that every catalogue system will be a target of each query. This means that the number of subqueries generated will be equal to the number of translators available to the Retrieval Manager.

Before we move on to developing the equation for the response time, we should note that the translation of the results and the aggregation of the results are likely to take far less time than most, if not all, of the other processes in Figure 4-2. With this in mind, we can drop them from consideration by making the following assumption:

Assumption 4.2.3: The time to translate and aggregate product descriptor search responses will be negligible compared to other processes and can be ignored for purpose of response estimation.

Using Equation 2.8, we can calculate the average response time for a local product descriptor search based on Figure 4-1. Additionally, we can use Equation 3.3 for the collection search time.

$$t_{local,prod} = t_{local,col} + N_{xlate} \times t_{sq} + t_{net-1}^{forward} + t_{xlate} + t_{net-2}^{forward} + t_{inv} + t_{net-2}^{reply} + t_{net-1}^{reply} \qquad \text{Eq. 4.2}$$

The time to form a single subquery is given as t_{sq}, while the time to translate a subquery is given as t_{xlate}. We define the network transmission times as t_{net}, with appropriate indicators as to which network they apply. The superscripts of t_{net} indicate whether it is a forward message (subquery) or the reply (response). N_{xlate} gives the number of unique translators available to the Retrieval Manager in question. Equation 4.2 can be simplified a bit further by assuming that the forward and reply times on the network are roughly equivalent and then combining those two terms. This assumption is reasonable so long as the two message sizes are not vastly different and the network throughputs are reasonable.

Since the translator times and the catalogue system times are likely to be more easily measured on a quiescent system, we should use the M/M/1 formula (q.v. Section 2.4.2) for those terms. Combining all this, we get the equation for the average response time for a local product descriptor search:

$$t_{local,prod} = t_{local,col} + N_{xlate} \times t_{sq} + {}^{d_{xlate}}\!\big/_{(1-\rho_{xlate})} + {}^{d_{inv}}\!\big/_{(1-\rho_{inv})} + 2 \times (t_{net-1} + t_{net-2}) \qquad \text{Eq. 4.3}$$

where d_{xlate} and d_{inv} are the average service times for the translator and the catalogue system, respectively, and ρ_{xlate} and ρ_{inv} are the utilization of the translator and the catalogue system, respectively.

Using the values of the parameters as listed in Table A-2, we find that the expected response time is approximately 125 seconds (2.1 minutes).

If Assumption 4.1.1 (persistent Z–association with translator) does not hold, then the time needed to form a Z–association, t_z, must be added to Equations 4.2 and 4.3. Equations 4.2 and 4.3 assume that the translator service time and response time include any time required to ready the translator if it is not already running.

Obiter Dictum: Equation 4.3 calculates the expected response time for a local product descriptor search in the general case. However, several points should be made for the benefit of those who wish to customize the equation for their particular site. In examining Figures 2-1 and

4-1, one notices that the individual subqueries traverse different networks, are handled by different translators, and are eventually serviced by different catalogue systems. Each of these will have their own distinct service (response) time. Equation 4.3 treats each member of the group as if it had the same response time. This is necessary in an analysis that attempts to cover a system as large as ICS. However, in order to use Equation 4.3 to estimate the response at an individual site, the slowest leg (subquery path) must be accounted for. One way of doing this is to use the maximum times for each of the networks, the translators, and the target catalogue systems in Equation 4.3.

4.3 On Multiple and/or Shared Translators

Assumption 4.2.1 tends to cause one to think in terms of each target catalogue system has it own translator (mathematically they are “one–to–one” and “on to”) and each translator communicates with only one Retrieval Manager. While it is intrinsically easier to implement a system this way than a system that shares translators or shares target catalogue systems, there are no inherent reasons why such sharing can not be done. Let us look at some of the implications of sharing translators or using multiple translators to access the same target catalogue system.

First, let us look at multiple Retrieval Managers accessing the same catalogue system, either with or without use of the same translator. The implication of doing this is that it is faster to share the catalogue system than it is to have collections that point to the appropriate terminal collection at a specific Retrieval Manager. This approach goes against the approach being developed by ICS and will not be considered further.

This leaves two approaches that should be investigated. The first is to use multiple translators, accessible by a single Retrieval Manager, to send queries to the same catalogue system. The other is to use a single translator to send queries to multiple catalogue systems (assuming that the target catalogue systems use the same protocol).

If we build a queuing diagram of the first case, ignoring networks, we get the following diagram:

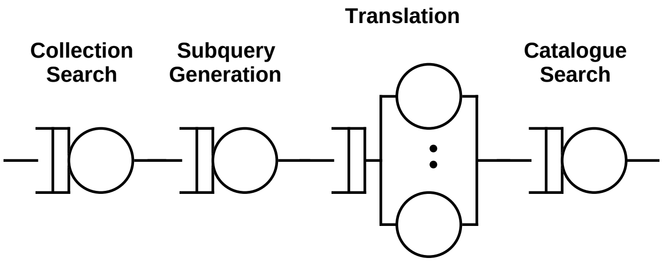

Figure 4-3. Using Multiple Translators to Send Queries to the Same Catalogue System

Looking at this diagram one comes to the realization that multiple translators are useful only if the translator is the bottleneck devices. The forced flow law (queuing theory) would indicate that the translator will be the bottleneck if and only if a) the translator time is larger than the transaction interarrival time, or b) its response time is larger than any of the collection search, subquery generation, or catalogue search times. While this condition is a disjunction (or), one can treat it a conjunction (and). The reason for this is that while condition “b” makes the translator a bottleneck, it makes little sense to improve its performance if it is exceeding the demand that is being put on it by condition “a”.

Now let us look at the second case, the use of a single translator to reference multiple catalogue systems from a single Retrieval Manager. This configuration could be considered as a way to contain costs associated with multiple translators doing the same protocol. Again we create a queuing network diagram:

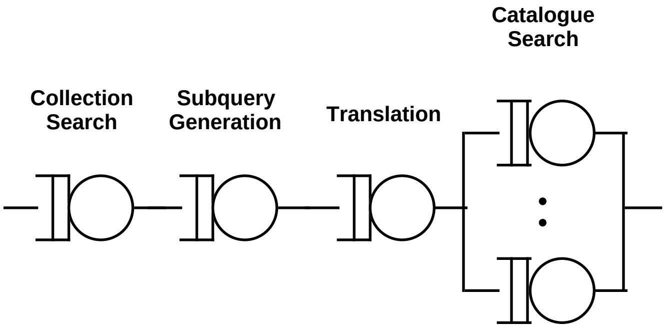

Figure 4-4. Use of a Single Translator to Reference Multiple Catalogue Systems from a Single Retrieval Manager

Again, we have to ask what are the conditions under which the translator would be the bottleneck device. As in the case above, the forced flow law (queuing theory) would indicate that the translator will be the bottleneck if and only if a) the translator time is larger than the transaction interarrival time, or b) its response time is larger than any of the collection search, subquery generation, or catalogue search times. Since there are multiple catalogue systems, if the load is balanced between them, there is a higher probability that the catalogue search will not be the bottleneck. However, if one considers the response time of some existing catalogue systems, it is by no means certain that the catalogue search will not be a bottleneck. Even if it is not, condition “a” still must be met together with the translator service time being higher than both the collection search and subquery generation times.

It should be noted that if the mapping of translators to catalogue search systems is one–to–many, then there must be a mechanism that will allow the Retrieval Manager to inform the translator which catalogue system is the target.

Implication: The use of the same catalogue system as a target of multiple Retrieval Managers is ruled out by the ICS approach. The use of multiple translators makes sense only if it can be shown that the translator is the slowest device (server). At the current time it does not seem likely that the translator will be the slowest server. The final implication is that one or more translators used to translate the same protocol to a bank of catalogue servers requires a method of indicating to the translator which catalogue system is the target of each subquery.

4.4 Sensitivity Analysis

In order to better understand the parameter space in which $t_{dist,col}$ is located, a sensitivity analysis was performed. The nominal values of the parameters are listed in Table A-2. For the purposes of the sensitivity analysis, it was assumed that all network speeds (throughputs) were identical.

When Equation 4.3 is examined several things become apparent. The first is that we expect the same sort of behavior of the response time for the same values of the equivalent parameters for the translators and inventory (catalogue) systems. Examining Table A-2, we find that expected service time of the catalogues is two orders of magnitude larger than that of the translators. Clearly we do not have to do a detailed sensitivity analysis of the translators parameters. Additionally, when we compare the other response times ($t_{local,col}$, t_{sq}, and t_{net}) with the service time of the catalogue systems, we find that they, too, are being dominated by the catalogue system. As a consequence, the only sensitivity analysis graphs we will present are for the parameters for the catalogue systems. The first chart (Figure 4-5) shows the sensitivity to the catalogue system utilization.

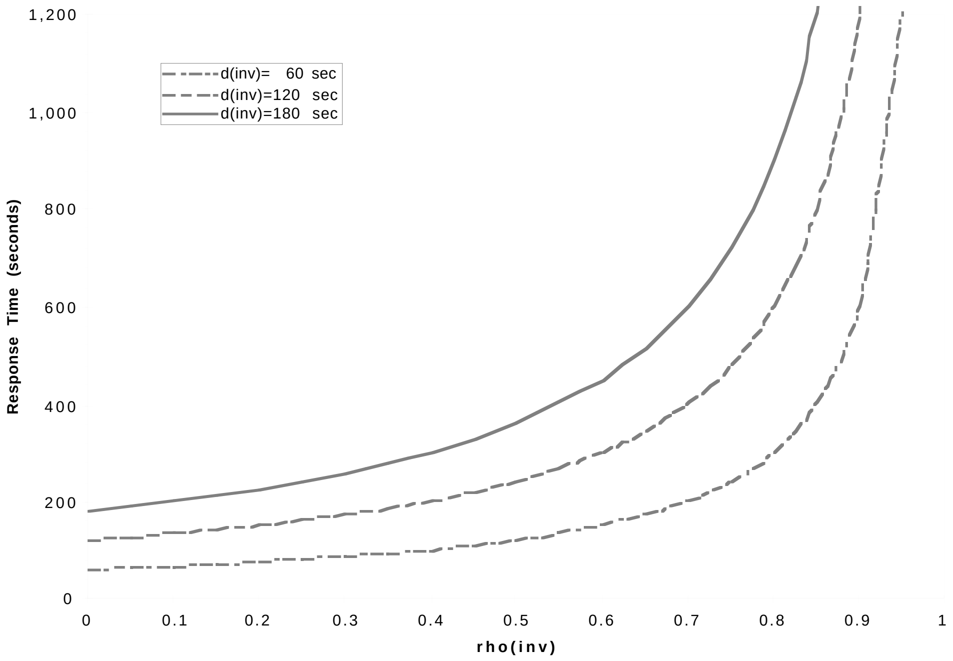

Figure 4-5. $t_{local,prod}$ vs. Catalogue Utilization

Figure 4-3 shows that one wants to hold down the utilization of the catalogue system. Additionally, one would like to have less utilization on the slower systems, a situation that does not seem likely to happen. Since ICS does not have direct control of the catalogue systems, another way to state the implications of this chart is to say that the response time of the ICS will not be driven by any of the ICS components, rather it will be driven by the underlying catalogue systems. This conclusion is reinforced by the plot of ICS response time vs. the catalogue system service time (Figure 4-6).

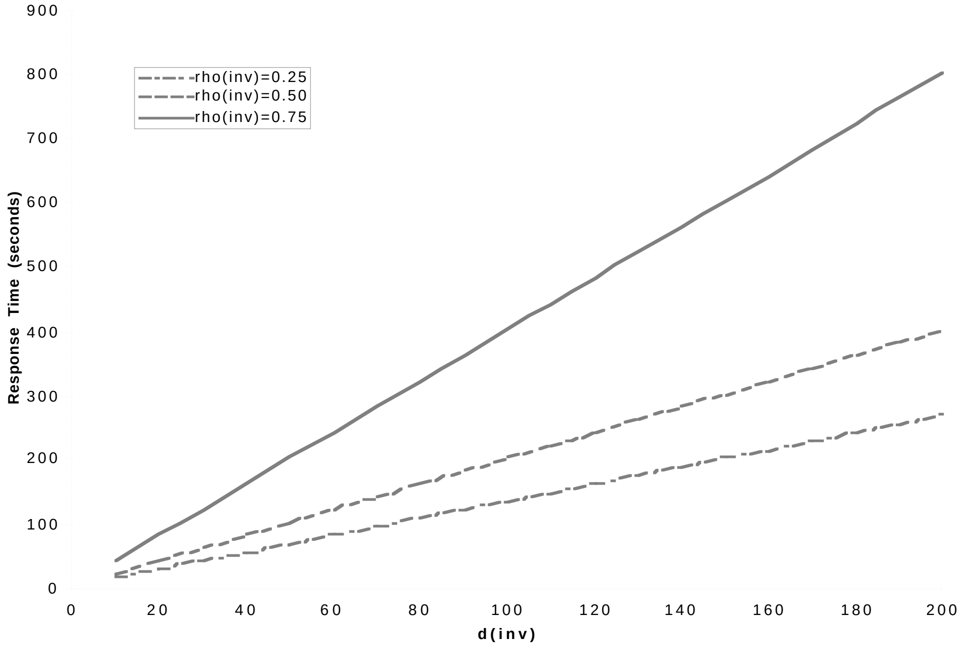

Figure 4-6. $t_{local,prod}$ vs. Catalogue System Service Time

While it was deemed redundant to reproduce the Figure 3-1 in this section, it should be remembered that $t_{local,col}$ forms part of the equation for $t_{local,prod}$. The interested reader can refer back to Section 3.4 to see the sensitivity exhibited by $t_{local,col}$. The point being that it must be kept in mind that the performance of the Retrieval Manager, specifically the collection search performance, can have a negative impact on the local product descriptor search response time.

This page intentionally left blank

5. Distributed Collection Search Response Time

5.1 Introduction

In a distributed collection search, a collection search query to a Retrieval Manager accesses at least one collection that is linked to (points to) a collection held by another Retrieval Manager. For any links of this type, a subquery has to be formed by the first Retrieval Manager and passed to the second one. At the second Retrieval Manager, the subquery may reference another collection that is remote and the process is repeated. This will continue until all remote links have been resolved. This process is shown diagrammatically in Figure 5-1.

Assumption 5.1.1: It is assumed that Retrieval Managers working on subqueries that were developed from the same initial query operate in parallel.

Assumption 5.1.2: Retrieval Managers pass subqueries to their targets as the subqueries are formed rather than waiting until all subqueries are formed.

These assumptions allow us the take advantage of the parallelism inherent in the ICS. In essence they state that a Retrieval Manager, having broken down a query into subqueries destined for multiple Retrieval Managers, will not send out the first subquery and wait for its response before sending out the next subquery.

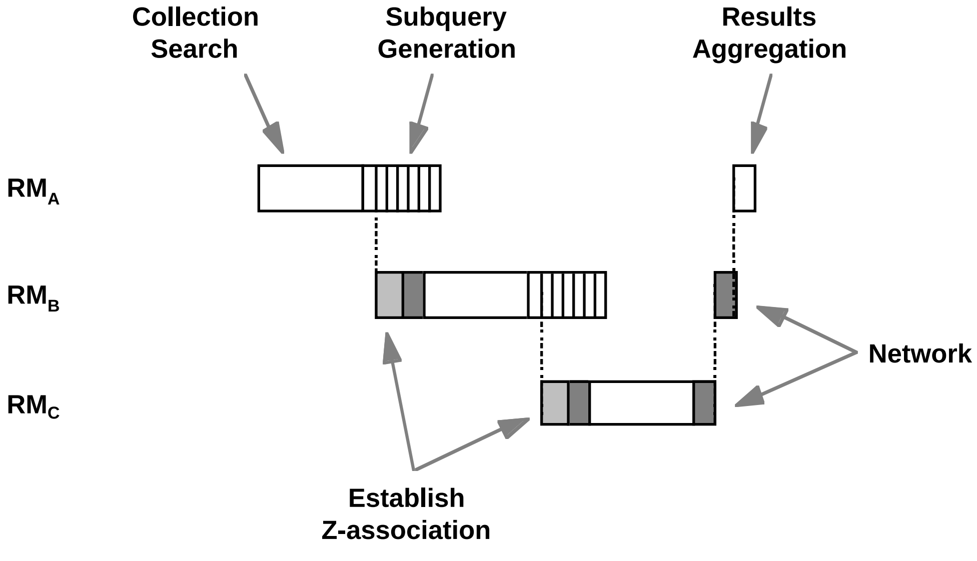

Figure 5-1. Distributed Collection Search vs. Time

The reader is also reminded of Assumption 4.2.1—there will only be one subquery formed per target Retrieval Manager, regardless of the number of target collections held by that Retrieval Manager. It should be mentioned that Assumption 4.2.2 allows us to say that the time to form the Z–associations, for those Retrieval Managers which will operate in parallel, is overlapped. Figure 5-1 attempts to show this by showing the z-associations being formed as the subqueries are developed. This is a fairly sensitive assumption—the response times will increase dramatically if this assumption is not valid.

5.2 Remote Links

5.2.1 Some Observations on Remote Links

In preparation for discussing distributed queries, we need to develop equations for the number of "remote" links that can be expected in a single search. Some clarification of the use of the word "remote" is in order. There are two senses in which one can call an object (e.g., a collection or a product) remote. The first implies the item is outside the CIP domain. In this sense of the word, product descriptors and products are remote even when they are held at the same site as the Retrieval Manager. The second sense of remote is referenced to the CEOS network. An object is remote if and only if it requires message traffic across the CEOS network in order to manipulate that object. The latter sense of remote is the one throughout this report.

Consider the case of a collection held by a Retrieval Manager (referred to as RM_A) containing a link to a collection held by a second Retrieval Manager (RM_B) and this collection or a subcollection links back to RM_A. Such a situation is shown in Figure 5-2. Technically speaking, this link back to RM_A is not a remote link, since it is on the original node. However, since we are after an estimation of the network traffic, we will consider both the $RM_A \rightarrow RM_B$ and the $RM_B \rightarrow RM_A$ links as being remote links.

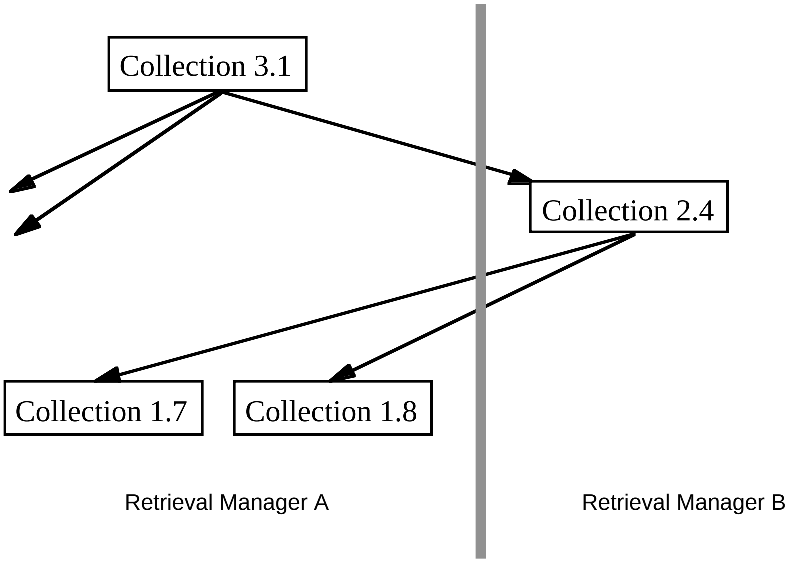

Figure 5-2. Remote Collections that Back–Map to Original Retrieval Manager

Now not every link will be a remote link, so we have to introduce a factor to account for that fact. We define R to be ratio of remote links to the total number of collection–to–collection links. The links between terminal collections and product descriptors are not included in the total. It should be noted the R is developed globally, that is, over all collections held by all Retrieval Managers. It is the number of all remote links throughout the ICS divided by the total number of links throughout the ICS. This definition of R does not place any restrictions on an individual collection. A single collection may reference product descriptors, only local collections, only remote collections, or a mix of local and remote collections.

Let us look for a moment at the range that R may take. If a collection is a provider archive collection, the value of R is zero (0), since an archive collection cannot include item descriptor members that are not under the direct control of the same Retrieval Manager that owns the physical collection. By contrast, it is quite possible to imagine user theme collections whose R is unity (1). A situation analogous to this exists on the World Wide Web (WWW) where there are sites that act as clearinghouses or otherwise point only to other sites. Obviously, ICS will not contain only provider archive collections or only user theme collections. All this really indicates is that R will be somewhere between zero (0) and one (1).

Recommendation: It is recommended that the PTT perform a Delphic study in order to estimate R. This study will use R=0.5 (i.e., one half the links are external) unless otherwise indicated.

In a distributed search, a search query to a Retrieval Manager accesses at least one collection that is linked to (points to) a collection held by another Retrieval Manager. For any links of this type, a subquery has to be formed by the first Retrieval Manager and passed to the second one. At the second Retrieval Manager this process can be repeated. Theoretically, this process could repeat itself at each subsequent Retrieval Manager until all collections within ICS have been accessed.

One would like to believe that a query would visit all collections, not all possible collections in ICS. However, references CIP, SDD, and URD all concur that it is not feasible to enforce the consistency of collection branches within a theme by the use of automated software. They then go on to state that it is the responsibility of the Retrieval Manager administrators to ensure theme consistency. As ICS grows and evolves and different individuals serve terms as administrators, different connections will be noticed and added between collections. As a consequence, it is difficult to determine *a priori* state how many different collections will be eventually reached from a given collection.

5.2.2 Depth of Link Traversal

Examining Figure 5-1, we see that we can determine the response time of a distributed query by summing the response times of longest chain of links. What we now need to do is to develop the expected number of links in the longest chain. The number is referred to as the depth of link traversal.

An individual link being remote or not forms a Bernoulli trial with probability R. We can now use the binomial distribution to calculate the probability that a collection will point to at least one remote collection.

$$p(\mathrm{l} \geq 1) = \sum_{x=1}^{n} P_n(x) = 1 - P_n(0) \qquad \text{Eq. 5.1}$$

The definition of $P_n(x)$ is given by:

$$P_n(x) = \binom{n}{x} \times p^x \times (1-p)^{n-x}$$

By definition the average number of collections pointed to by a collection is A (q.v. Section 2.3.2) and the probability of a link being external is R (q.v. Section 5.2.1). We know that there is some possibility that one or more of the external links will point to the same external collection. So we need to correct for collection–to–collection overlap (O_{col}, q.v. 2.3.2). Also, not every external link will be the target of the search. We introduce a factor, *h*, to account for the probability that a given collection will fulfill a query (be the target of a search). We can now update the calculation of the probability that at least one link will be remote:

$$p(\mathrm{l} \geq 1) = 1 - (1 - \mathrm{R})^{h \times (1 - \mathrm{O}_{col}) \times \mathrm{A}} \qquad \text{Eq. 5.2}$$

Equation 5.2 allows us to relate the probability of a single link being remote (R) to the probability that any given collection will have a link to an external collection.

Because of Assumption 3.1.1, a Retrieval Manager’s collection structure is not searched one collection at a time. Together with Assumption 2.3.3, it causes us to interrogate a portion of the collection structure at a Retrieval Manager in one pass rather than one collection at a time. Therefore, we need to correct Equation 5.2 for this.

L_{RM} (q.v. Equation 2.7) corrects for the collection–to–collection overlap, but does not take into account that when a link is external it cannot contribute to visiting additional links within the Retrieval Manager. Rather than overload the L_{RM} parameter, we will define a new parameter, L_{CE}, that takes this into account. Its development is similar to that of L_{RM} in Section 2.3.4. The governing equation is:

$$L_{CE} = \sum_{k=1}^{\frac{D_{RM}}{2}} (1 - \mathrm{R})^{k-1} \times \left(1 - \mathrm{O}_{col}\right)^{k} \times \mathrm{A}^{k}$$

This equation has the closed form of:

$$\sum_{k=1}^{n} x^{k-1} y^{k} = \frac{y - x^{n} y^{n+1}}{1 - xy}$$

Substituting we get:

$$L_{CE} = \frac{\left(1 - \mathrm{O}_{col}\right) \times \mathrm{A} - (1 - \mathrm{R})^{D_{RM}/2} \times \left(1 - \mathrm{O}_{col}\right)^{(D_{RM}/2)+1} \times \mathrm{A}^{(D_{RM}/2)+1}}{1 - (1 - \mathrm{R}) \times \left(1 - \mathrm{O}_{col}\right) \times \mathrm{A}} \qquad \text{Eq. 5.3}$$

Now returning to Equation 5.2:

$$p(\mathrm{l} \geq 1) = 1 - (1 - \mathrm{R})^{\mathrm{h} \times \mathrm{L}_{\mathrm{CE}}} \qquad \text{Eq. 5.4}$$

For a remote query, we can no longer consider the collection structure within only a single Retrieval Manager. We now turn our attention to the process of spawning subqueries that occurs during a remote query. As mentioned in Section 5.2.1, when we traverse a remote link we may interrogate a collection that also contains a remote link. This process will continue until we end up at a collection that contains no remote links. This last collection may be a terminal collection.

This type process gives rise to a geometric distribution whose probability, p, is given by

$$\mathrm{p} = 1 - p(\mathrm{l} \geq 1)$$

The mean of the geometric distribution is 1/p, so we get an average number of links traversed in a chain (subtracting one (1) since we are counting links, not collections):

$$L_D = (1 - \mathrm{R})^{-(\mathrm{h} \times \mathrm{L}_{\mathrm{CE}})} - 1 \qquad \text{Eq. 5.5}$$

The geometric distribution assumes that you will continue the process indefinitely until you eventually arrive at a collection that contains no remote collections. In ICS this process cannot continue indefinitely since the collection structure is finite. We will deal with the termination of the process a bit later.

At this juncture it is appropriate to look at the growth of the depth of link traversal as a function of various parameters. The average depth of link traversal given by Equation 5.5 uses L_{CE} which is given by Equation 5.3. Figures 5-3 through 5-7 graphically represent the relationship in Equation 5.5 for several different parameters. The vertical line in the graph indicates the nominal value of the parameter on the x–axis. Each graph contains three (3) curves. The middle curve in each graph is the nominal value for the parameter listed in the legend. As can be seen from the curves, the growth in depth of link traversal becomes larger as each of the parameters A, D, h, and R increases and as $\mathrm{O}_{\mathrm{col}}$ decreases. Each time an external link is traversed a Z–association is formed. If the time to form a Z–association, t_{z}, is relatively large and the depth of traversal is large, the time to perform a distributed query can become significant. The value of t_{z} for this study is one half minute. If we try to keep the distributed collection search time less than that of the catalogue (inventory) system, then the depth of link traversal must be four (4) or less. The reason for this desire will become obvious in Section 6.

The reader is reminded that due to the parameters being varied, the total number of collections underlying the plots in each of the graphs may be different. The figures were supplied not to make absolute comparisons, but to allow the reader to get a feel for which parameters are significant and how ICS may change as it evolves.

The statistics of the geometric distribution used to Equation 5.5 assumes that you will keep traversing links until you finally find no more remote links. This is obviously not the case with ICS, since there is a finite number of collections. There is a practical maximum to the traversal depth imposed by this. We will investigate what this maximum is in order to see if helps us. First however, we need to investigate how many Retrieval Managers are involved in each layer of a remote search.

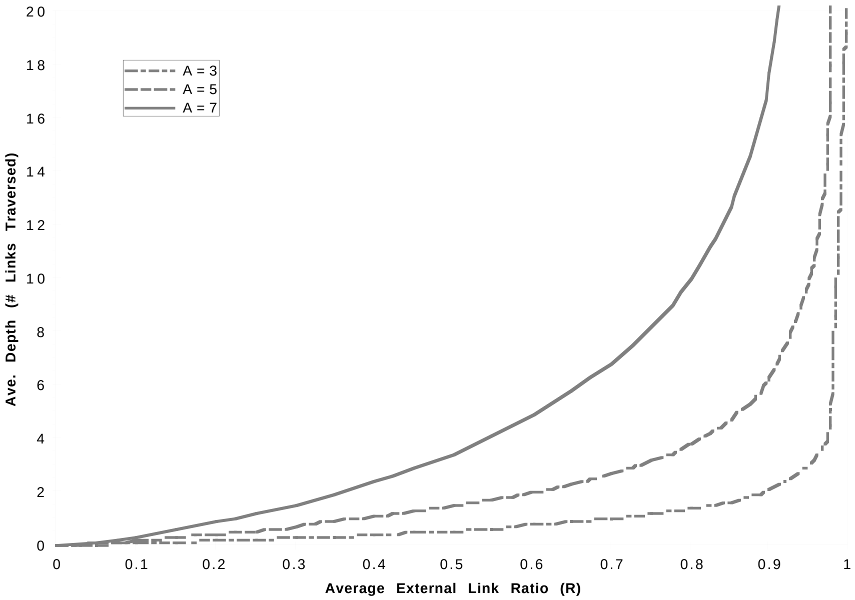

Figure 5-3. Average Depth of Link Traversal vs. R

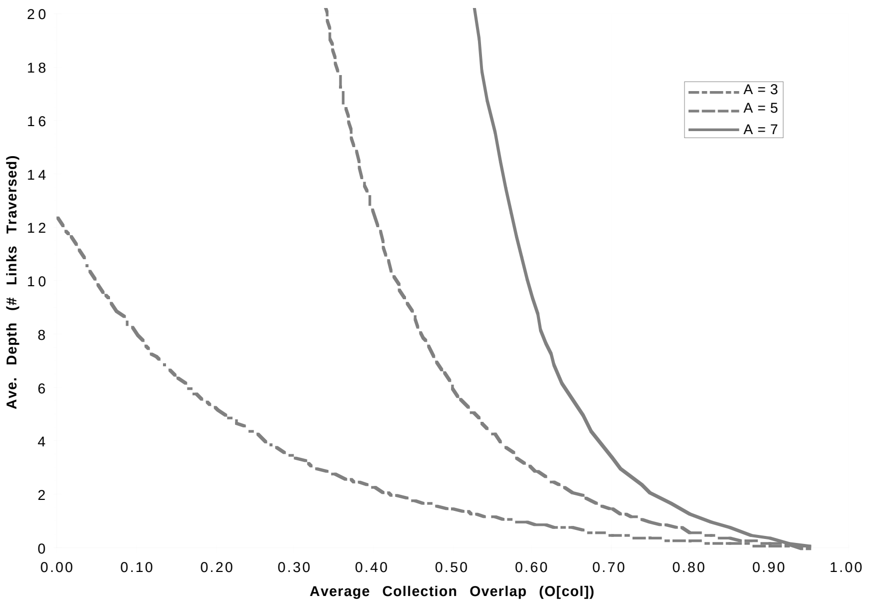

Figure 5-4. Traversal Depth vs. Average Collection Overlap (O_{col})

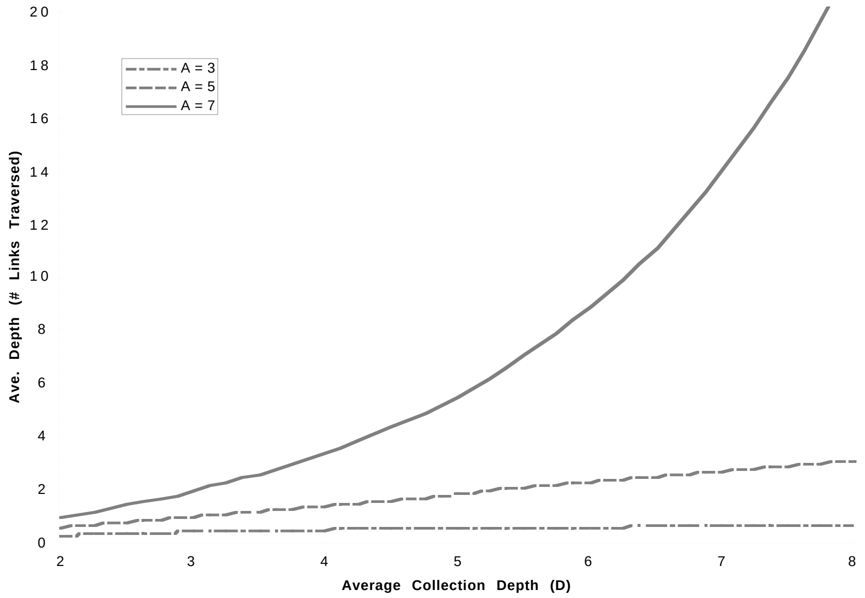

Figure 5-5. Traversal Depth vs. Average Collection Depth (D)

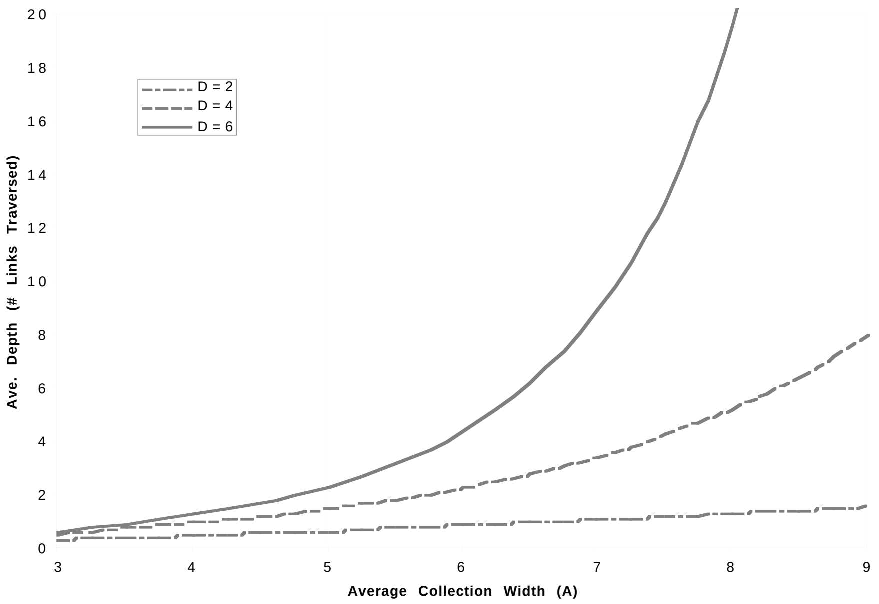

Figure 5-6. Traversal Depth vs. Average Collection Width (A)

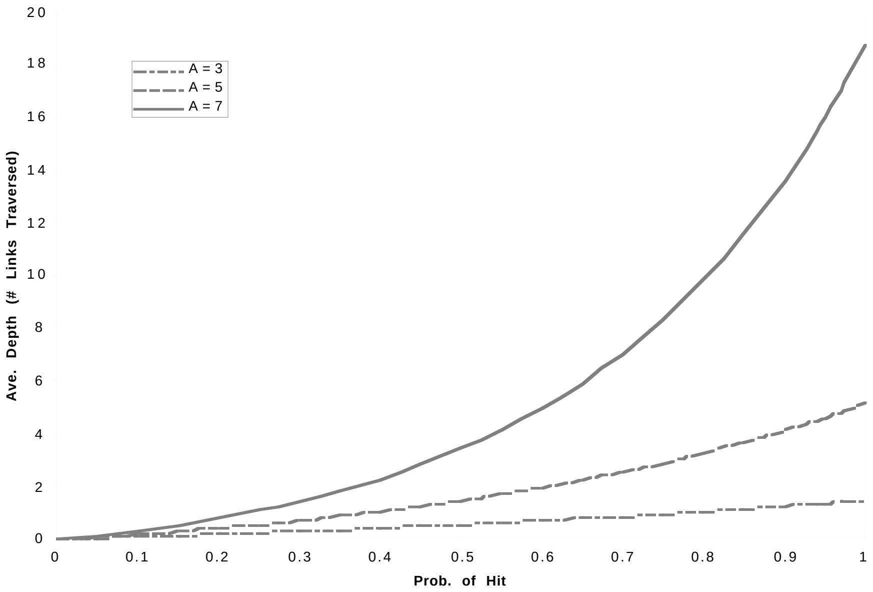

Figure 5-7. Traversal Depth vs. Search "hit" Probability

5.2.3 Width of Remote Collection Search

For reasons that will hopefully become obvious we wish to investigate the number of Retrieval Managers that will receive subqueries during a collection search. We know from Equation 5.3 total number of links that will be searched at a single Retrieval Manager (L_{CE}). We also know that a fraction of these, R (q.v. Section 5.2.1), will contain links to other Retrieval Managers. However, it is quite possible that several of these links may be directed to the same remote Retrieval Manager. Finally, we know that only a fraction, h, of the collections will satisfy the initial query.

Stated formally, in response to a query there are $h*R*L_{CE}$ remote subqueries being sent to $N_{sites}-1$ Retrieval Managers. We wish to know what is the average number of Retrieval Managers that will receive subqueries. Because there are multiple Retrieval Managers we cannot use one of the more common distributions. In fact, because the subqueries are mutually exclusive (i.e., a subquery goes to only one Retrieval Manager) and exhaustive (i.e., every subquery goes to some Retrieval Manager), it would appear that the distribution is a multinomial.

Rather than to use the multinomial distribution we will use combinatorial analysis. The probability that exactly *m* Retrieval Managers do not receive a subquery is given by:

$$\rho_m = \binom{N_{sites}-1}{m} \times \binom{\lfloor h\times R\times L_{CE}\rfloor -1}{N_{sites}-m-2} \Big/ \binom{N_{sites}+\lfloor h\times R\times L_{CE}\rfloor -2}{\lfloor h\times R\times L_{CE}\rfloor} \qquad \text{Eq. 5.6}$$

The reader is reminded of the following definitions regarding binomial coefficients:

$$\binom{x}{0} = 1 \qquad\qquad \binom{x}{r} = 0, \quad \text{if either } r > x \text{ or } r < 0.$$

The reader is also warned that binomial coefficients are often defined strictly in terms of factorials. This can lead one to believe that the upper index of the binomial coefficient must always be an integer. This is not the case. In fact, the upper index does not even have to be a positive number. The lower index, on the other hand, must be an integer. We are, however, using a combinatorial interpretation of the binomial coefficients. As a consequence, both indices are restricted to nonnegative integers. Examining the upper index of the second term we realize that this implies that the product $h*R*L_{CE}$ must be greater than one (1). The practical interpretation of this is that we must be dealing with an average at least one subquery. It should also be noticed that the product $h*R*L_{CE}$ will, in all likelihood, be noninteger. The conservative approach is to take the floor function of the product, as was done in Equation 5.6.

Since we are dealing with the average we recall that, for a random variable *X*, assuming values $x_1, x_2, \ldots$ with corresponding probabilities $p(x_1), p(x_2), \ldots$

$$Average(X) = E(X) = \sum_k x_k \times p(x_k) \qquad \text{Eq. 5.7}$$

Using Equations 5.6 and 5.7 we get L_W, the average number of Retrieval Managers that will receive subqueries directly from the initial query.

$$L_W = \sum_{k=0}^{N_{sites}-2} \left(N_{sites} - 1 - k\right) \times \rho(k) \qquad \text{Eq. 5.8}$$

Examination of Equations 5.6 and 5.8 will disclose the use of a factor of N_{sites}–1. The reason for this is that we assume that a subquery will not go to the Retrieval Manager that generated it (q.v. Assumptions 2.3.2 and 3.1.1). The reason that N_{sites}–2 is used as the index in the summation of Equation 5.8 is that it is not possible to have a subquery that does not go to one of the Retrieval Managers, coupled with the fact that we must exclude the generating Retrieval Manager.

The formulae behind L_W are complicated enough that, as we did for L_D, we will look at the width of link traversal (number of Retrieval Managers contacted) as a function of various parameters. Figures 5-8 through 5-13 graphically represent the relationship in Equation 5.8 for several different parameters. The vertical line in the graph indicates the nominal value of the parameter on the x–axis. Each graph contains three (3) curves. When the calculations were performed it was noticed that the lowest curve became uninteresting (typically it became zero). To avoid this the O_{col} parameter was set to 0.5, rather than 0.7 as used elsewhere in this study. Most of the graphs show a quantization induced by the use of the floor as described above.

The reader is reminded that, due to the parameters being varied, the total number of collections underlying the plots in each of the graphs may be different. The figures were supplied not to make absolute comparisons, but to allow the reader to get a feel for which parameters are significant and how ICS may change as it evolves.

Figure 5-8 shows a roll–off at high values of R. This is due to the fact that so many links are remote that the total number of links at the initial Retrieval Manager drops.

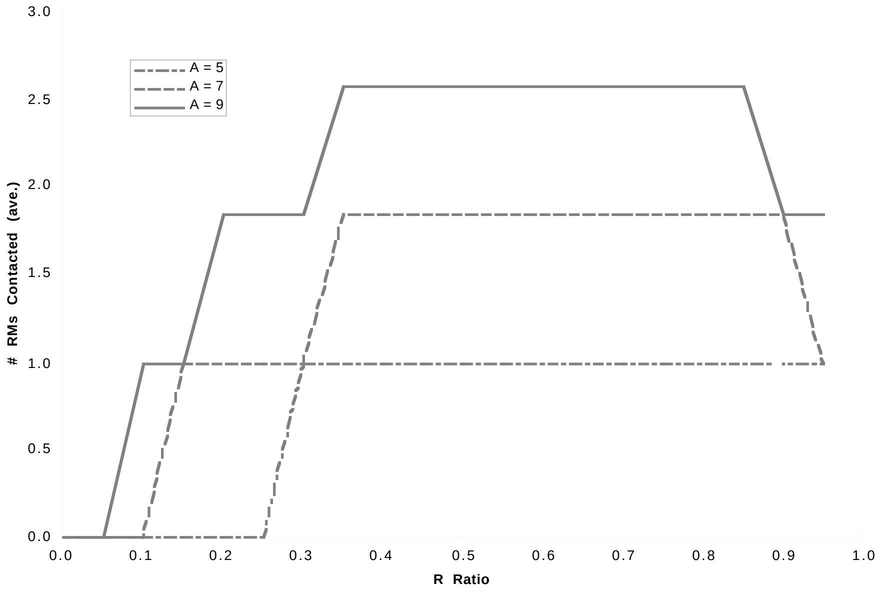

Figure 5-8. Average Width of Traversal vs. R

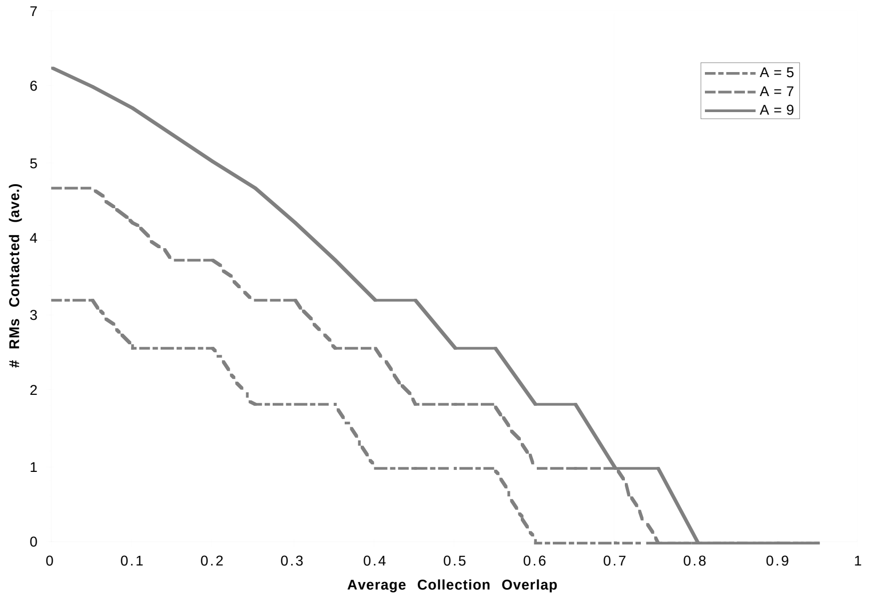

Figure 5-9. Traversal Width vs. Collection Overlap (O_{col})

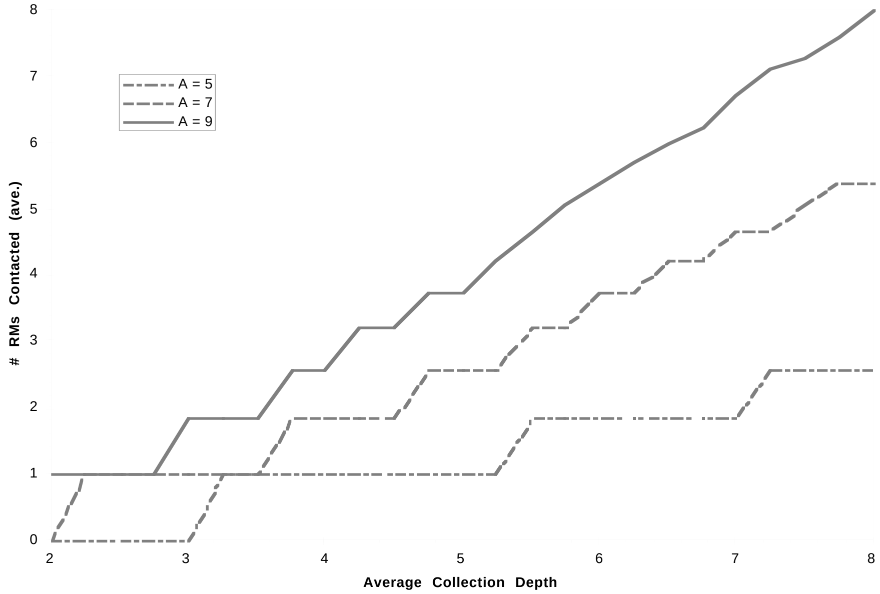

Figure 5-10. Traversal Width vs. Average Collection Depth (D)

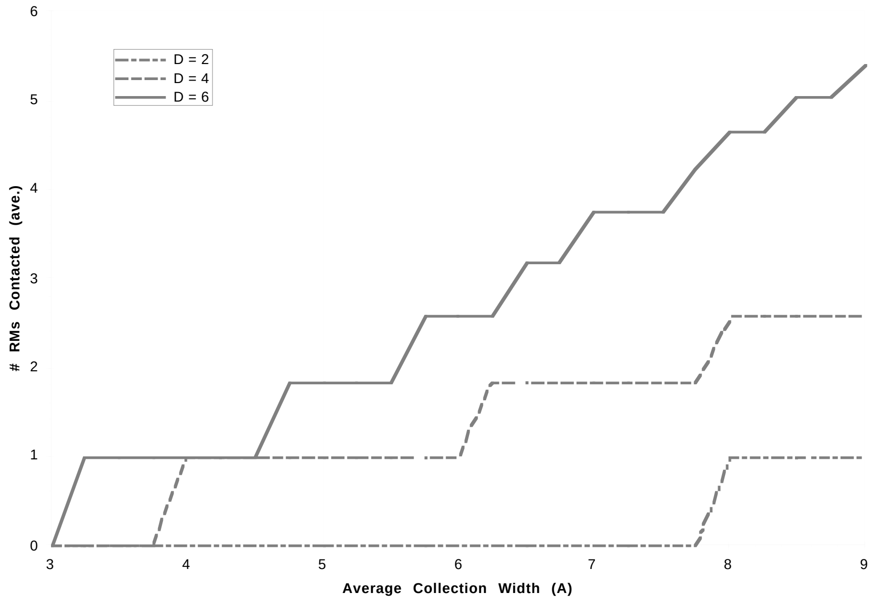

Figure 5-11. Traversal Width vs. Average Collection Width (A)

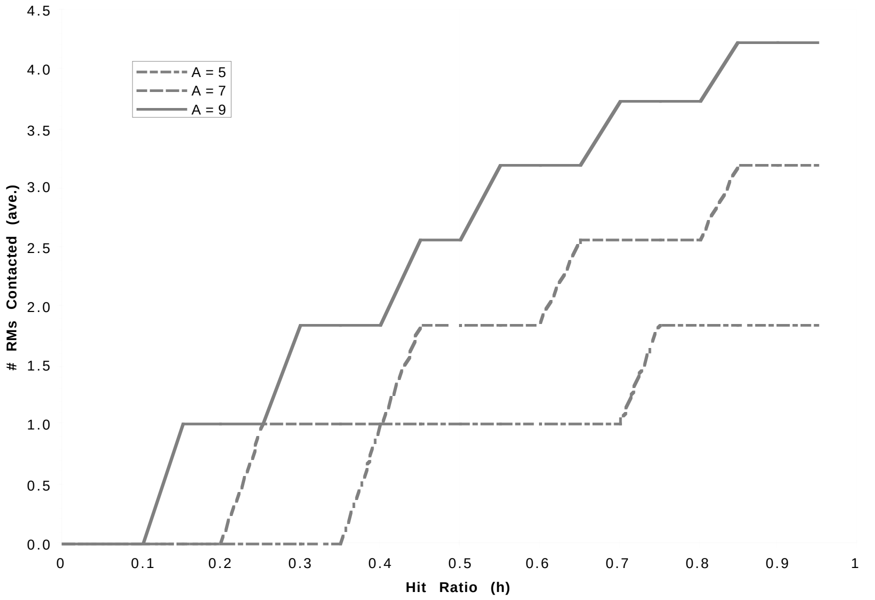

Figure 5-12. Traversal Width vs. Search “hit” Probability

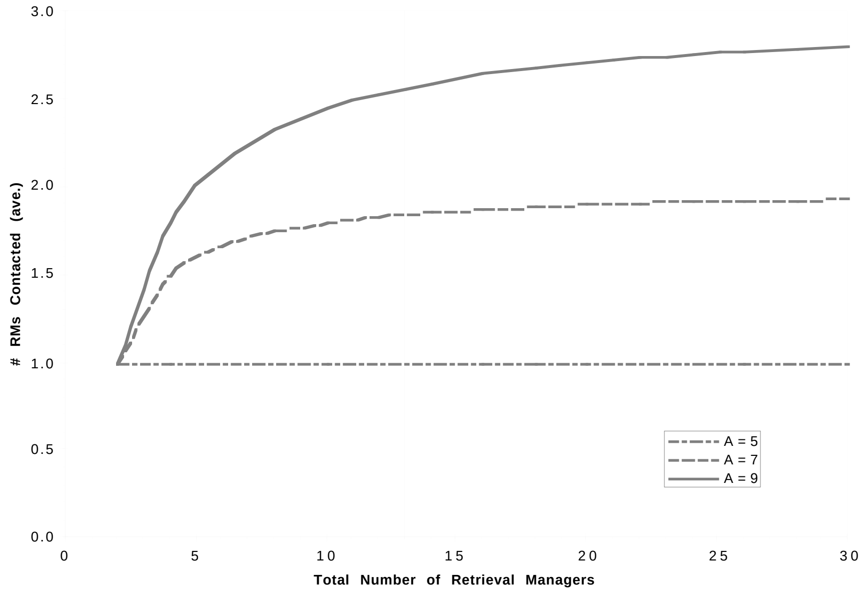

Figure 5-13. Traversal Width vs. Total Number of Retrieval Managers

5.2.4 Depth of the Global Collection Structure

Up to this point we have occasionally treated the collection structure as a uniform tree, even though technically it isn’t. For local searches, either collection or product descriptor, this is not particularly far off. It is possible that in some cases the collection structure departs from a uniform tree, but our calculations of local search query response times do not require the collection structure to be a uniform tree, so there is no real problem.

This situation becomes different when we calculate the average number of links traversed, L_D, in a distributed search. Here chains of links that revisit a Retrieval Manager such as those shown in Figure 5-2 can have an impact. What we find when we have distributed searches is that, while our calculations are not sensitive to whether or not the global collection structure approaches a uniform tree, it is sensitive to the global collection structure depth, D_G, and to the average depth of traversal during a distributed query (q.v. Equation 5.5). We have seen that the average number of links traversed can grow rapidly. What we now need to do is to look at how D_G is estimated.

Figure 5-14 shows two relative simple collection structures interacting via external links to form a deeper global collection structure. Each individual collection structure has four collections at three levels, giving a depth of 2 for each structure. However, the way their external links are combined results in the global collection structure having a depth of 4. In fact, had the link from Collection 2.4 to Collection 1.1 been reversed, the global collection structure would have had a depth of 5. More complicated global collection structures that have a greater depth could have been created from the same two local collection structures.

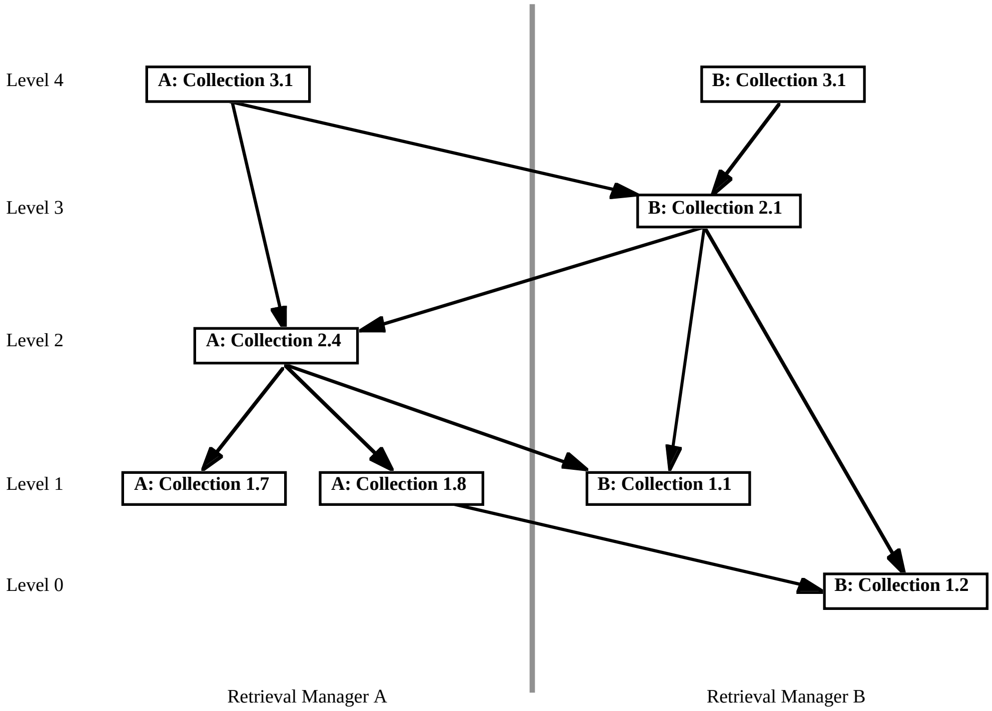

Figure 5-14. External Links in the Global Collection Structure

Figure 5-14 does show a link from Collection 1.8 to Collection 1.2, so the collections labeled 1.x can not be terminal collections. One can imagine the terminal collections being below the 1.x collections. The intent of the diagram is to indicate that external links can cause relatively simple local collection structures to become a complex global collection structure.

Some thought about this situation leads us to the realization that the worst case occurs when the external links cause us to visit each node in global collection structure. We can then calculate the worst case for D_G with the equation:

$$D_G = \left(\sum^{\forall \text{sites}} \sum_{k=1}^{D_{RM}} A^k \right) - 1$$

This equation sums from k=1, rather than k=0, because the terminal collections cannot possess external links. If we use the average depth, D, rather than the depth of the collection structure at each individual Retrieval Manager in this equation, we get:

$$D_G = \left(N_{sites} \times \sum_{k=1}^{D} A^k \right) - 1 \qquad \text{Eq. 5.9}$$

If we estimate D_G using the values we have been using in the study, namely A=5, D=4, and N_{sites}=13, we get D_G = 13 * 156 – 1 = 2,027. The geometric distribution term of Equation 5.5 forces the average L_D to be 1.5 if half the links are remote (i.e., R=0.5). This is a fairly large mismatch between the worst case depth and what we think is the average search depth. We need to consider why this is.

The first answer is that the worst case global tree structure should not occur. Proper attention by the Retrieval Manager administrators to the commonality of links within a collection should keep the global collection depth to a reasonable value. There is something more subtle as well. Some thought about how one would go about constructing theme collections reveals that the overlap parameter, O_{col}, varies considerably over the collection structure hierarchy. At the global collection and root collection levels, there will be, in all likelihood, no overlapping links. In the level or two above the terminal collections, the overlap may be fairly high.

Mathematically handling this situation is beyond the scope of the present study. One could conceive of lowering the value of O_{col} used in Equation 5.3 and hence in Equations 5.5 and 5.8. This is not a perfect solution, but would allow one to use the equations developed in this study. In the long term, however, if one desires to develop better predictive equations, a study looking at this should be undertaken.

Recommendation: It is recommended that the PTT perform additional study on how collection–to–collection overlap varies within the collection structure hierarchy.

We now return to the consideration of the depth of the global collection structure. Let us consider the situation where there are two Retrieval Managers each with its own administrator. We will refer to the Retrieval Manager as RM_A and RM_B and the administrators are RMA_1 and RMA_2. Let us assume that the Retrieval Managers each contain a collection that shares a common theme. Graphically, the situation is represented as Figure 5-15.

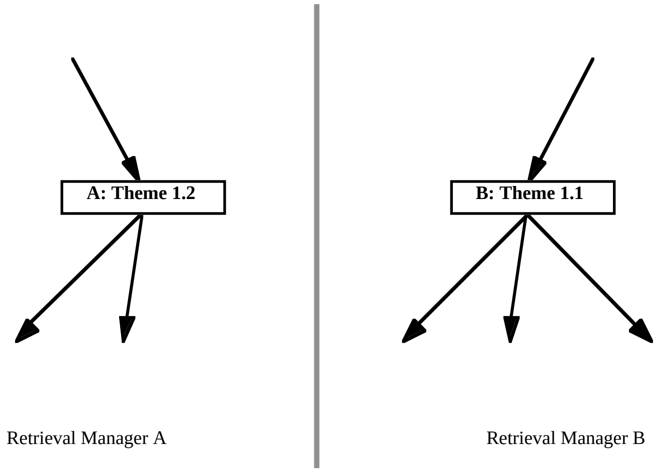

Figure 5-15. Two Retrieval Managers with Separate but Related Themes

If RMA_1 examines the appropriate collection (Theme 1.1) in RM_B and decides that it contains sufficient commonality with his, he is free to add a remote link into his collection (Theme 1.2). This is shown as the solid line connecting Theme 1.2 to Theme 1.1 in Figure 5-16. Unfortunately in doing so, he has precluded RMA_2 from making a similar link from Theme 1.1 to Theme 1.2. This is indicated as the dashed line in Figure 5-16. The reason that this link cannot be made is that it would form a cycle (loop).

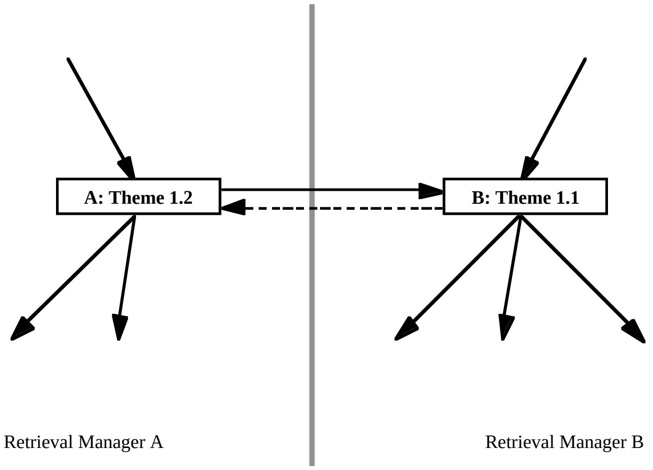

Figure 5-16. Attempt to Link the Related Theme Collections

RMA_2 can always choose to link the Theme 1.1 collection to the various lower level collections pointed to by Theme 1.2. For that matter, RMA_1 could have linked Theme 1.2 to the collections pointed to by Theme 1.1. There are some drawbacks in going this route. One of the drawbacks is that RMA_2 would need to perform more collection maintenance. As an example, if RMA_1 added another link to Theme 1.2, RMA_2 would have to discover this and add the same link into Theme 1.1. As ICS grows and more datasets are added at the various Retrieval Managers, one can imagine that this form of link maintenance could become one of the major tasks of an administrator.

There is at least one way around this situation. That is to create additional collections that are not necessarily visible to the ordinary user of ICS. For lack of any better terminology, we will refer to collections visible to ordinary users as public and those that visible only to Retrieval Manager administrators as private. It should be noted that these distinctions and mechanisms to create them are not necessarily present in the URD or the CIP specification. Figure 5-17 shows how the public/private collections would be used in the case presented above. New collections visible to ordinary users have been added. These collections are labeled Public 1.1 and Public 1.2. These collections are used to reference the two private collections Theme 1.1 and Theme 1.2. This setup allows each Retrieval Manager administrator to maintain his own collection structure without having to be aware of whether administrators at other Retrieval Managers are adding or removing collections.

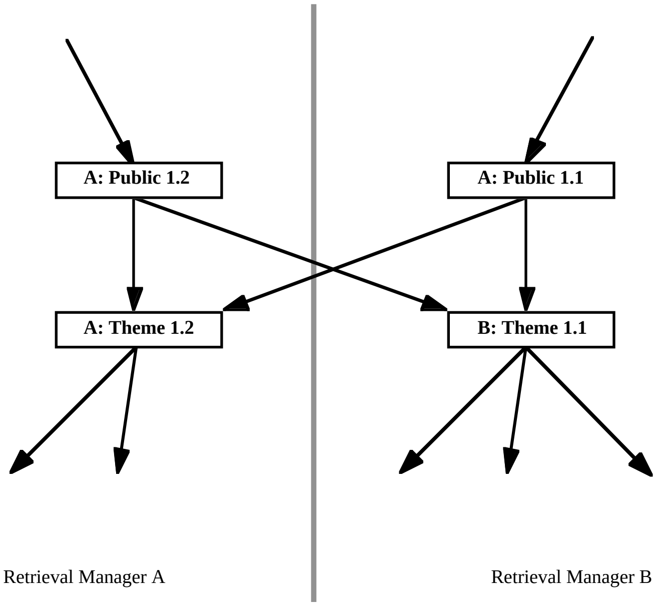

Figure 5-17. The Use of Public and Private Collections

If such a mechanism was added to ICS, it would result in less maintenance work by the Retrieval Manager administrators.

Recommendation: It is recommended that a public/private distinction together with appropriate support mechanisms be added to ICS.

The discussion of public/private collections also shows us a method of controlling the depth of the global tree. This method is to allow collections to have remote links only to collections that are lower than themselves on the collection structure hierarchy. All the links in Figures 5-2 and 5-17 obey this rule. This rule seems to be the way that one would tend to build collections in the first place. The public/private distinction also facilitates this rule.

Recommendation: It is recommended that CEOS adopt the convention that collections may be linked only to collections lower than themselves on the collection structure hierarchy.

With the adoption of these two rules the worst case depth of the global collection structure becomes much more tractable. One would never want to put a public collection above either the root collection or a terminal collection. Given this the worst case for D_G becomes:

$$D_G = 2 \times D - 1 \quad \text{Eq. 5.10}$$

5.2.5 Limits to Remote Collection Traversal

Let us put aside the issue of the depth of the global collection structure for a while. Because of Assumption 2.3.3, we can say that the average query comes in at the D/2 level of a Retrieval Manager's collection structure. Assuming a uniform tree, the fraction of the collection structure that is accessed by the average query is:

$$F = \frac{1}{1 + \left(1 - \mathrm{O}_{col}\right)^{\mathrm{D}/2} \times \mathrm{A}^{\mathrm{D}/2}}$$

We will call the fraction of the collection structure a subtree. The fraction means that a reasonable maximum for the number of subtrees during a distributed search is:

$$n_{st} = N_{sites} \times F^{-1} = N_{sites} \times \left(1 + \left(1 - \mathrm{O}_{col}\right)^{\mathrm{D}/2} \times \mathrm{A}^{\mathrm{D}/2}\right) \qquad \text{Eq. 5.11}$$

This maximum does not indicate whether they are visited one after another in a sequential fashion or whether visited in a highly parallel fashion. However, after some thought we should notice that the subtrees form the same relationship to the global collection structure as does a local search to the local collection structure.

Having realized this, we should also realize that the math developed in Section 2.3.2 also applies. The number of subtrees visited on a distributed search that has depth L_D and width L_W is:

$$T_{st} = \sum_{k=0}^{L_D} \left(L_W\right)^k = \frac{1 - \left(L_W\right)^{L_D + 1}}{1 - L_W} \qquad \text{Eq. 5.12}$$

If we have an estimate for the average number of different Retrieval Managers that will be contacted in a subquery (L_W), we can use Equations 5.11 and 5.12 to determine the depth at which, on the average, every collection within the global collection structure will have been examined.

$$L_D(\max) = \frac{\ln\left(N_{sites} \times \left(1 + \left(1 - \mathrm{O}_{col}\right)^{\mathrm{D}/2} \times \mathrm{A}^{\mathrm{D}/2}\right) \times \left(\left(1 - \mathrm{O}_{col}\right) \times L_W - 1\right) + 1\right)}{\ln\left(\left(1 - \mathrm{O}_{col}\right) \times L_W\right)} - 1 \qquad \text{Eq. 5.13}$$

It should be noted that if the product (1-O_{col})*L_W is not greater than one (1) there is some probability that a link will not be formed and therefore the maximum depth has no real meaning. In those cases where the product (1-O_{col})*L_W is equal to one (1), L_D(max) is equal to n_{st}. L_D(max) was calculated for different values of L_W using A=5, D=4, Nsites=13. The results of this calculation are shown in Figure 5-18. Four different overlap (O_{col}) values are shown. There are two things to be noted about this graph. The first is that as percentage of overlap increases, the wider the fan out L_W has to be. Otherwise, the probability of revisiting a collection becomes too high and you need more depth to examine the necessary number of subtrees. The other thing to notice is that as you approach this minimum width the curve becomes asymptotic.

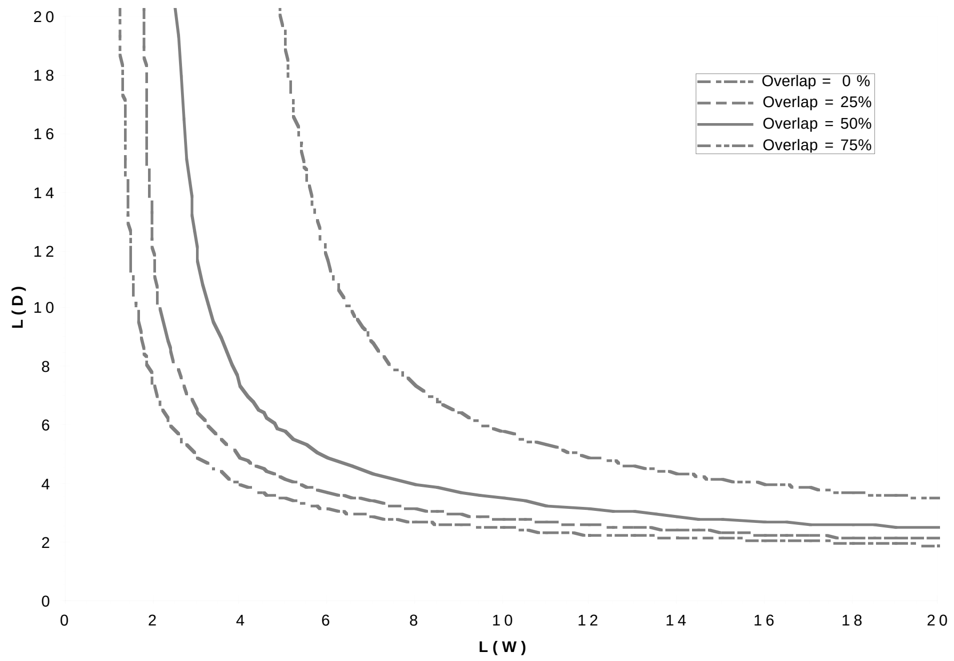

Figure 5-18. Average Search Depth vs. Average Search Width

We have now developed a probability–based formula for calculating the search depth (q.v. Equation 5.5) and two limiting conditions, the global collection depth (q.v. Equation 5.10) and subtree depletion (q.v. Equation 5.13). In the next section where we develop the equations for the average response time of a distributed collection search we will use limited depth. For reference, we express this as:

$$L_D(\mathrm{lim}) = \mathrm{minimum}\left(L_D, D_G, L_D(\mathrm{max})\right) \qquad \text{Eq. 5.14}$$

5.3 Nominal Response

The steps required to perform a distributed collection search are as follows:

1. perform the initial query,
2. form subsequent subqueries,
3. form Z–associations with the targets of the subqueries,
4. pass the subqueries to the target Retrieval Managers,
5. the target Retrieval Managers repeat the process, and
6. the targets pass their results back.

The time required for Step 1 we know from the discussion in Section 3.3 and is $t_{local,col}$. In Step 3 we need to be concerned only with the number of unique Z–associations that are formed. From the geometric distribution we know that the average number of unique Z–associations will be L_D. If we define t_z to be the average time required to form a single Z–association, we can calculate the time required for Step 3. For Step 2 we define the time required to form a single subquery as t_{sq}. Recalling Assumption 4.2.1, we know that the number of subqueries will also be the same as the number of Z–associations.

Steps 4 through 6 are recursive, but from Section 5.2 we know that the depth of the recursion is L_D(lim). For now we will define a function, $t_{recur}(\mathsf{W},\mathsf{l})$, for the time required for each recursion. The first argument, W, gives the average number of unique Z–associations that will be formed at each level of the recursion. The second argument, l, gives the number of recursions that are desired. Subsequently, we will refine $t_{recur}(\mathsf{W},\mathsf{l})$.

At this point the time required for a distributed collection search is given by:

$$t_{dist,col} = t_{local,col} + t_{recur}\left(L_W, L_D(\lim)\right) \qquad \text{Eq. 5.15}$$

In refining the value of $t_{recur}(\mathsf{W},\mathsf{l})$ we note that it contains the same six (6) steps listed above, plus network time to and from the target Retrieval Manager. We will define the network transmission times as t_{net}, with superscripts to indicate whether it is the forward message or the reply. It should be noted at this point that we, via Assumption 4.2.2, assume that the Z–associations are formed in parallel. If we use this information to develop the equation for $t_{recur}(\mathsf{W},\mathsf{l})$ we get:

$$t_{recur}(\mathsf{W},\mathsf{l}) = \mathsf{W} \times t_{sq} + t_Z + t_{net}^{forward} + t_{net}^{reply} + t_{local,col} + t_{recur}(\mathsf{W},\mathsf{l}-1)$$

Because of form of the equation, we can express this function without the recursion:

$$t_{recur}(\mathsf{W},\mathsf{l}) = \mathsf{l} \times \mathsf{W} \times t_{sq} + \mathsf{l} \times \left(t_Z + t_{net}^{forward} + t_{net}^{reply} + t_{local,col}\right)$$

This means Equation 5.15 becomes:

$$t_{dist,col} = L_D(\lim) \times \left(L_W \times t_{sq} + t_Z + t_{net}^{forward} + t_{net}^{reply} + t_{local,col}\right) + t_{local,col} \qquad \text{Eq. 5.16}$$

Using the values of the parameters as listed in Table A-2, we find that the expected response time is approximately 50 seconds.

5.4 Sensitivity Analysis

In order to better understand the parameter space in which $t_{dist,col}$ is located, a sensitivity analysis was performed. The nominal values of the various parameters are given in Table A-2. For the purposes of the sensitivity analysis, it was assumed that the request and reply transit times across the network were identical.

As the analysis was performed, it became obvious that many of the parameters were sensitive variables (i.e., small changes in their values lead to large changes in the response time). In many cases the maximum response time was limited by the limits on remote collection traversal (q.v. Section 5.2.5 and particularly Equation 5.14). Figure 5-19 is an example of this. The graph plots the response time versus the average number of collections per collection (A) for three (3) different values of the collection depth (D). The response time levels off for high A on two of the curves due to the global collection depth being completely searched (the D_G parameter has become the limiting factor in Equation 5.14.

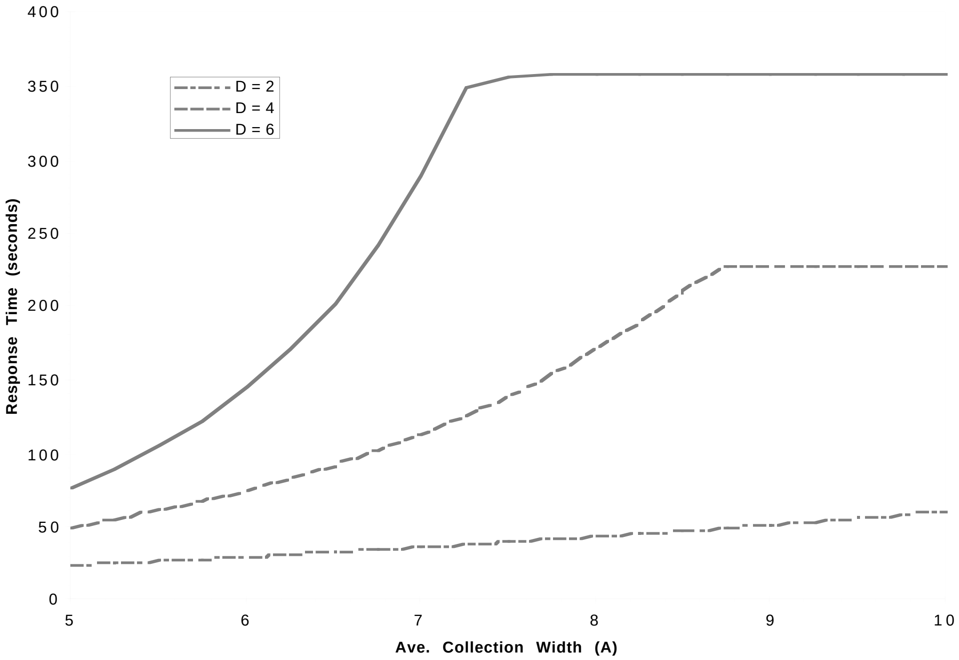

Figure 5-19. $t_{dist,col}$ vs. Average Collection Width (A)

Figure 5-20, the plot of response time versus the collection–to–collection overlap illustrates each of the three (3) factors in Equation 5-14. Each of the curves in Figure 5-20 shows the same effects, although it may be easier to talk about it in the A=9 case. In the first portion of the curve (from O_{col} = 0.0 to approximately 0.35) the $L_D(max)$ term is the dominate one. This indicates the global collection structure is being pretty much completely searched. The flat top (from 0.35 to about 0.7) is being dominated by the global collection depth term, D_G. Finally, the smoothly varying downslope (O_{col} = 0.75 and higher) is being dominated by the L_D term. This means search is being damped statistically by the probability of linking to a collection that has already been examined.

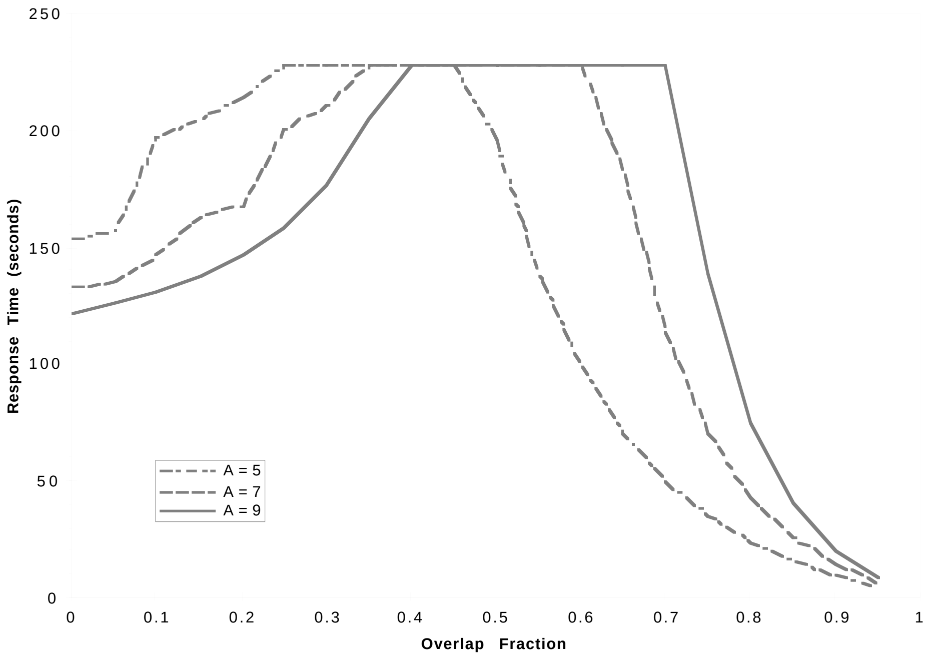

Figure 5-20. $t_{dist,col}$ vs. Collection–to–Collection Overlap (O_{col})

Figure 5-21 shows the response time as a function of the ratio of external links to the total number of collections (this ratio is R). The higher values of R mean that more Retrieval Managers will be sent subqueries. This increases the probability of long search chains. At some point the global depth is exceeded and the search chain cannot increase any further.

Rather than to show graphs for each parameter involved in the basis functions, the decision was made to show only a subset of the possible graphs. If the reader is interested in the general trend of a graph, rather than absolute numbers, then he can consult some of the plots in Section 6. As a case in point, Figure 6-2 shows the distributed product descriptor search analogue of Figure 5-21. Figure 6-2 gives the results for two (2) different values of t_z, whereas Figure 5-21 gives the results for three (3) different values of A. The trend to be noted in Figure 6-2 is that differences in t_z matter more at higher R values. Because of the similarity in the basis functions, this relationship holds true for the distributed collection search as well.

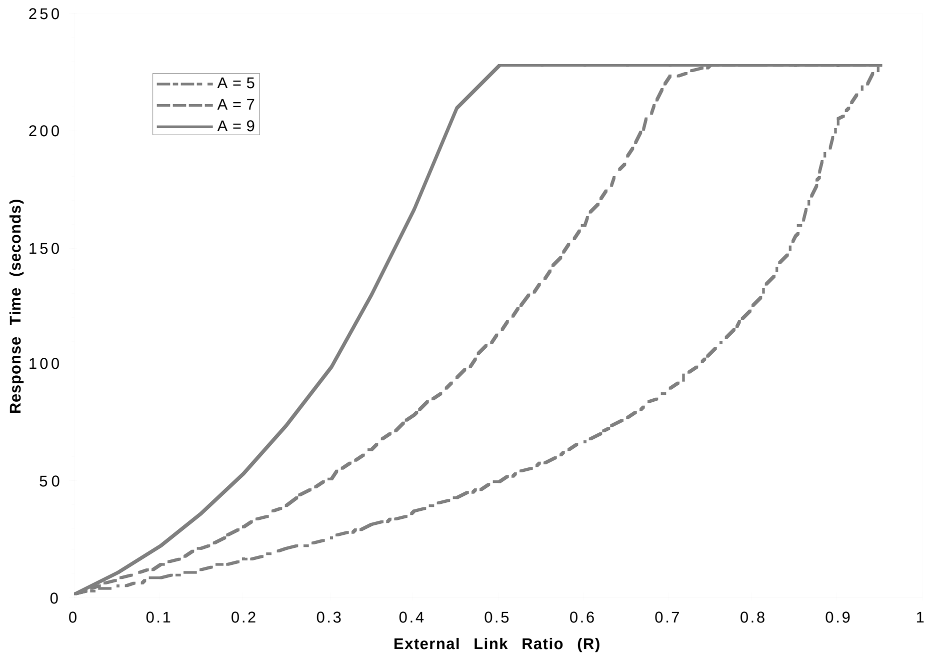

Figure 5-21. $t_{dist,col}$ vs. the External Link Ratio (R)

Figure 5-22 gives the relationship between the time required to establish a Z–association and the overall response time. When we examine the slope of the curves in Figure 5-22 we can see that there is a fair leverage there. That is, lower t_z will have a fair effect on the overall response time. Figure 6-3 is the distributed product descriptor search analogue of Figure 5-22. It was plotted at three (3) values of R, rather than three (3) values of A. A and R have similar affects on the response time, but A has larger effect.

If one were to go back and examine the governing equations, Equations 5.3, 5.5, 5.8, and 5.16, one would notice a high degree of interaction among the various parameters. This tends to indicate that each Retrieval Manager will have its own characteristics due to the set of collections that it is responsible for. Given the sensitivity of some of the parameters, this may make the system hard to tune.

The distributed collection search response times, as estimated by this study, are marginal–to–adequate, depending on the end–user's expectations. However, the response time estimates are sensitive to multiple parameters, increasing the risk to CEOS. As a consequence, two recommendations are made. One of the features of ICS and the collection structure that may not be adequately treated by a strictly statistical (probability) approach, such as the one used in this study, is how different Retrieval Managers will be subqueried during a query of theme collections. If theme collections tend to be broad (high A) and shallow (low D), the average response time may drop. It is worth some additional study.

Recommendation: It is recommended that the PTT perform additional study with regard to the nature of queries against theme collections.

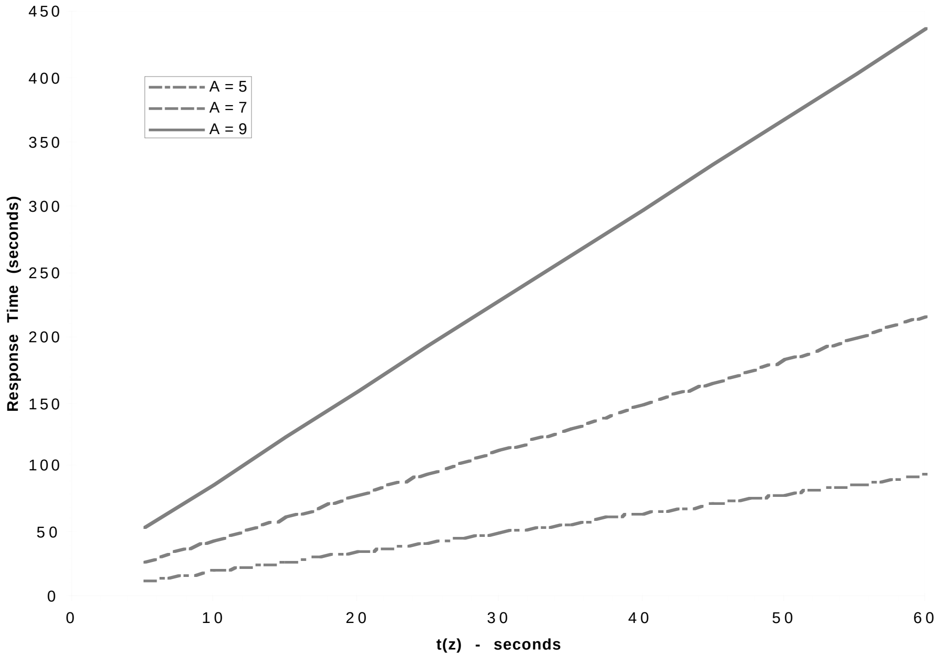

Figure 5-22. $t_{dist,col}$ vs. the Time to form a Z–association (t_z)

One of the reasons why distributed collection searches require a fair amount of time is the time required to form a Z–association. If this time could be lower or the number of Z–associations required could be lowered the response times would improve. For this study it was estimated that the Z–association time is 30 seconds. Given the speed of modern computers, it would be worth several million instructions of bookkeeping to avoid having to form a new z-association. Persistent Z–associations among Retrieval Managers would significantly improve the distributed search response time.

Recommendation: It is recommended that the PTT examine schemes to either lower or eliminate the time required to form a Z–association or to lower the number of Z–associations needed.

While it was deemed redundant to reproduce the Figure 3-1 in this section, it should be remembered that $t_{local,col}$ forms part of the equation for $t_{dist,col}$. The interested reader can refer back to Section 3.4 to see the sensitivity exhibited by $t_{local,col}$. The point being that it must be kept in mind that the performance of the Retrieval Manager, specifically the collection search performance, can have a negative impact on the distributed collection search response time. A little thought should make it clear that if a single Retrieval Manager is overloaded, or is slow for any other reason, the distributed collection search response time will be negatively impacted.

Recommendation: CEOS may wish to consider establishing a process of ongoing operational timing checks in order to determine which systems are becoming overloaded. This would allow CEOS to make sure that the ICS middleware would not become the slowest item in the response time.

Nota bene: If we examine Equation 5.16, we notice that there are several factors that have the same multipliers as does the time to form a Z–association (t_z). These factors are the network response times, (t_{net}), the local collection search response time, ($t_{local,col}$), and the time required to spawn a subprocess (t_{sq}). Any one of these factors could become the dominating parameter for distributed collection search response times if they become larger than any of the others. A traditional problem for many systems in the past has been the network response times. The reader is cautioned not to overlook the other contributors to response time if the ICS response times become large.

This page intentionally left blank.

6. Distributed Product Search Response Time

6.1 Nominal Response

The distributed product descriptor search is the distributed analog of the local product descriptor search. It can be considered as two separate phases, a distributed collection search followed by a product descriptor search. For the time required for the distributed collection search we can use Equation 5.4. Recalling Assumption 2.3.1, we know that product descriptor search will always be at the same Retrieval Manager as the terminal collection. This allows us to use Equation 4.3 for the product descriptor search phase. It should be noted that the local collection search portion of Equation 4.3 is subsumed into the distributed collection search phase. Combining the two phases, we get:

$$t_{dist,prod} = t_{dist,col} + N_{xlate} \times t_{sq} + {d_{xlate}}\Big/{\left(1-\rho_{xlate}\right)} + {d_{inv}}\Big/{\left(1-\rho_{inv}\right)} + 2 \times \left(t_{net-1} + t_{net-2}\right) \qquad \text{Eq. 6.1}$$

Using the values of the parameters as listed in Table A-2, we find that the expected response time is approximately 173 seconds (2.9 minutes).

6.2 Sensitivity Analysis

In order to better understand the parameter space in which $t_{dist,prod}$ is located, a sensitivity analysis was performed. The nominal values of the various parameters are given in Table A-2. For the purposes of the sensitivity analysis, it was assumed that the request and reply transit times across the network were identical. It was also assumed that all network speeds (throughputs) were identical.

Figure 6-1 shows the effect of catalogue system utilization on ICS. At the higher utilization, the effects are fairly extreme. This diagram is slightly misleading in that the utilization plotted is the average utilization across all the catalogue systems participating in the query. A single slow catalogue system may or may not be noticeable to the ICS user.

Figure 6-2 shows the response time as a function of the ratio of external links to the total number of collections (R). As one expects, the larger the value of R, the more widely dispersed are the collections involved in the query and, hence, the response time is worse. Figure 5-21 is the distributed collections search analogue of Figure 6-2. One can consult Figure 5-21 to get a better understanding of how the average number of collections per collection (A) and R interact. One should keep in mind that the trends of the distributed collection and product descriptor searches will be similar but not their actual values.

Figure 6-3 gives the response time as a function of the time required to form a Z-association. The response time increases linearly as a function of t_z. Figure 5-22 is the distributed collection search analogue of this graph.

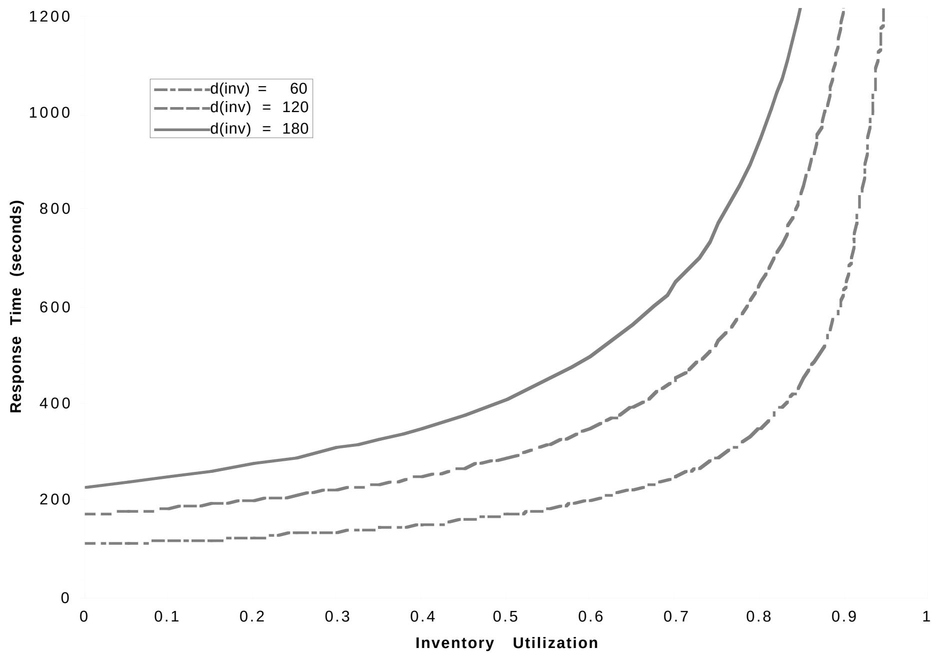

Figure 6-1. ***$t_{dist,prod}$* vs. Catalogue System Utilization (ρ_{inv})**

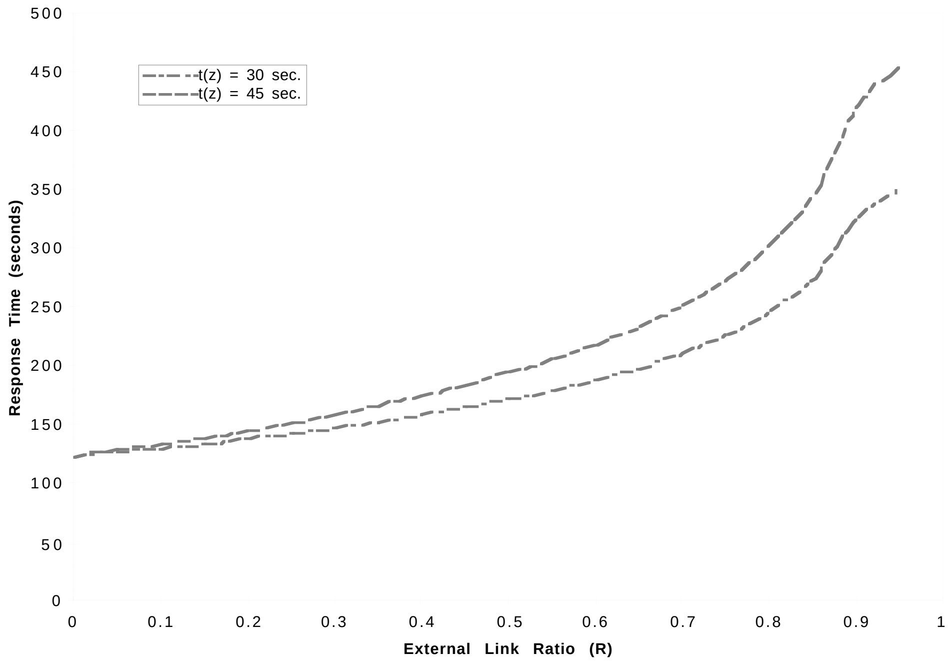

Figure 6-2. ***$t_{dist,prod}$* vs. the External Link Ratio (R)**

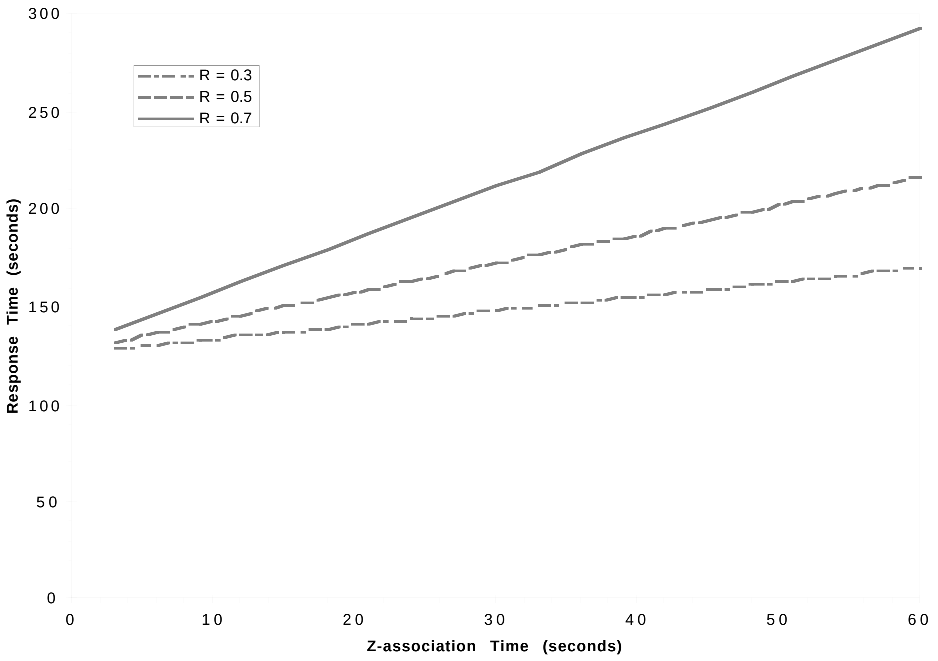

Figure 6-3. $t_{dist,prod}$ vs. the Time to form a Z–association (t_z)

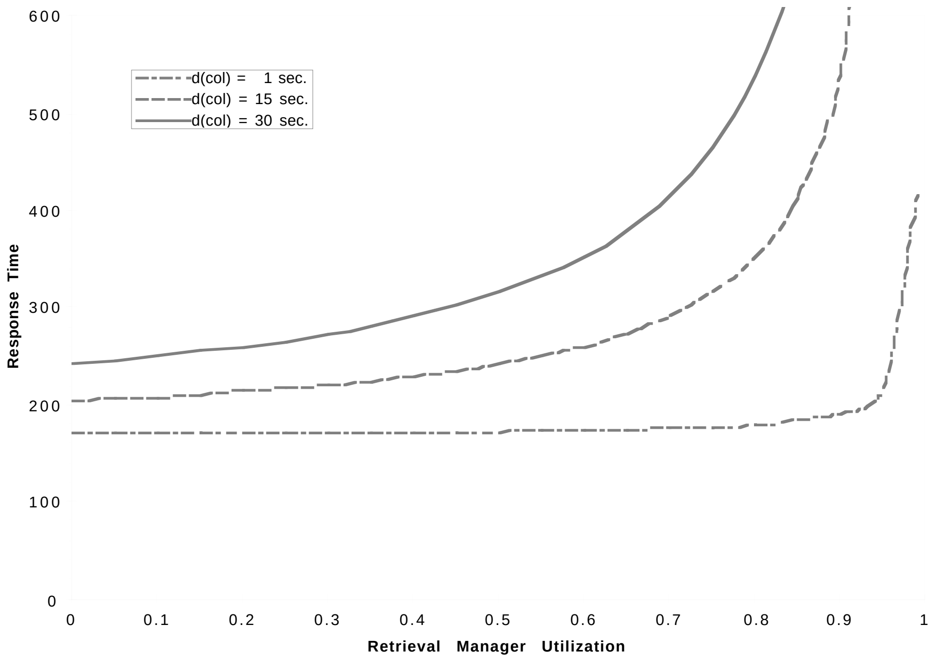

Figure 6-4. $t_{dist,prod}$ vs. Retrieval Manager Utilization (ρ_{RM})

While it was deemed redundant to reproduce the Figure 3-1 in this section, it should be remembered that $t_{local,col}$ forms part of the equation for $t_{dist,prod}$. The interested reader can refer back to Section 3.4 to see the sensitivity exhibited by $t_{local,col}$. The point being that it must be kept in mind that the performance of the Retrieval Manager, specifically the collection search performance, can have a negative impact on the distributed collection search response time. A little thought should make it clear that if a single Retrieval Manager is overloaded, or is slow for any other reason, the distributed collection search response time will be negatively impacted. Figure 6-4 is an attempt to show this effect, as it plots the response time against Retrieval Manager's utilization. Also included in Figure 6-4 is three (3) different values for the Collection Data Base (CDB) service time. Since the distributed product descriptor search has to do, in essence, a distributed collection search at multiple Retrieval Managers, the effect of increased CDB service time is dramatic.

Appendix A. Inputs and Assumptions

A.1 Inputs

The following table gives the definition and nominal value of the parameters used in this study.

Table A-1. Parameter Definitions

Symbol	Definition
A	Average number of collections per collection
D	Collection depth
d_{col}	Average collection query service time
D_G	Depth of the Global Collection Structure
d_{inv}	Average inventory (catalogue) query service time
d_{xlate}	Average catalogue translation service time
h	Probability that a collection will satisfy a query.
L_{CE}	L_D corrected for collection–to–collection overlap
L_D	Average number of external links traversed in a chain (depth of search)
L_D(lim)	L_D limited by various processes
L_D(max)	Maximum L_D before the collection structure is depleted (fully examined)
L_{RM}	Average number of links traversed during a query
L_W	Average number of external links examined in parallel
N_{sites}	Number of ICS sites (Number of Retrieval Managers)
N_{xlate}	Average number of unique translators at a Retrieval Manager
O_{col}	Probability of overlapping collections (Amount of Overlap)
O_{prod}	Probability of overlapping links to product descriptors
R	Ratio of remote links to total number of links
r_{inv}	Average number of product descriptors referenced by a local query
ρ_{inv}	Inventory catalogue system utilization
ρ_{RM}	Retrieval Manager utilization
r_{term}	Average number of terminal collections referenced by a local query
ρ_{xlate}	Catalogue translator utilization
$t_{dist,col}$	Average response time required for a distributed collection search
$t_{dist,prod}$	Average response time required for a distributed product descriptor search
t_{fill}	Average time to pass a message or response to the network
$t_{local,col}$	Average response time required to search a local collection
$t_{local,prod}$	Average response time required to perform a local product descriptor search
t_{net}	Network service time (q.v. Section 2.5)
T_{RM}	Average number of collections traversed (visited) in a query
t_{sq}	Average time required to form a subquery from a query
t_{xmit}	Average response time of network transmission
t_z	Average time required to establish a Z–association

The following table gives the units, range, and nominal value for the parameters used in this study. Parameters not listed in the table are derived within the report.

Table A-2. Parameter Values and Ranges

Symbol	Units	Nominal	Range	Source
A	n/a	5	5-50	Delphi
D	n/a	4	2-7	Delphi
d_{col}	seconds	1		Logica
d_{inv}	seconds	60	12-240	Delphi
d_{xlate}	seconds	1		Hughes/1
h	n/a	0.5	0-0.99	Hughes/2
N_{sites}	n/a	13	10-200	CINTEX Sites
N_{xlate}	n/a	5		Hughes/2
O_{col}	n/a	0.7	0-0.95	Logica
R	n/a	0.5	0-0.95	Logica
ρ_{inv}	n/a	0.5	0-0.99	Hughes/2
ρ_{RM}	n/a	0.5	0-0.99	Hughes/2
ρ_{xlate}	n/a	0.5	0-0.99	Hughes/2
t_{fill}	seconds	0.01		Logica
$t_{local,col}$	seconds	5		URD
t_{sq}	seconds	0.01		Logica
t_{xmit}	seconds	0.16		Germain
t_z	seconds	30	0-60	Logica & Hughes/1

Legend:

Delphi	PTT Delphic study, see Appendix B.
Germain	Estimate by Andy Germain based on “ping” of current networks.
Hughes/1	Estimate based on translator prototype.
Hughes/2	Estimate supplied by Hughes Information Technology Systems.
Logica	Estimate supplied by Logica.
URD	Requirement 374 in reference URD.

A.2 Assumptions

The assumptions used within the document are listed below for ease of reference. In order see to the rationale for the assumptions it is necessary to consult the text where the assumptions are introduced. Assumptions are numbered with three (3) digits. The first and second digits give the section and subsection numbers where the assumption can be found. The third digit is a sequence number within the subsection. As an example, Assumption 2.3.1 would be the second assumption in Section 2.3.

Assumption 2.2.1: When a Retrieval Manager contacts another Retrieval Manager on behalf of a user (e.g., in order to perform a search), a Z–association is formed. It is not terminated until the user closes his session with the original Retrieval Manager.

Assumption 2.2.2: Piggybacking will be requested on a minimal number of the searches and therefore its effects can be ignored. Pre–staging of results is not performed.

Assumption 2.3.1: Product descriptors (and, by implication, product data) are not directly accessed over the CEOS network. They are held by catalogue systems that are accessible by the Retrieval Manager, albeit possibly through intervening networks and translators.

Assumption 2.3.2: For performance estimation, we assume that during searches Retrieval Managers are efficient in detecting and ignoring collection–to–collection overlap. A result of this assumption is that a Retrieval Manager will not search a collection more than once for a given local collection search.

Assumption 2.3.3: For searches originating with a human user, the distribution of the collection levels of the searches will be uniform across all levels of the collection hierarchy.

Assumption 2.3.5: For purposes of analysis, it is assumed that a collection at the ith level is referenced only by collections at the (i+1)th level. In turn, it references collections at the (i–1)th level.

Assumption 3.1.1: It is assumed that within a given Retrieval Manager the collection database has been well laid out and is fairly efficient with regard to the expected queries.

Assumption 4.1.1: The Retrieval Manager forms persistent Z-association(s) with the catalogue translator(s) and geoserver(s) with which it cooperates, i.e., no INITs are required during a search operation.

Assumption 4.2.1: The Retrieval Manager will generate a single subquery for each distinct Retrieval Manager, translator, and/or geoserver that is a target of the query. This subquery may reference multiple collections within the target server.

Assumption 4.2.2: The time to fork individual subclients that send subqueries to Retrieval Managers, translators, and target systems is negligible.

Assumption 4.2.3: The time to translate and aggregate product descriptor search responses will be negligible compared to other processes and can be ignored for purpose of response time estimation.

Assumption 5.1.1: It is assumed that Retrieval Managers working on subqueries that were developed from the same initial query operate in parallel.

Assumption 5.1.2: Retrieval Managers pass subqueries to their targets as the subqueries are formed rather than waiting until all subqueries are formed.

A.3 References

Aho *et al* — Data Structures and Algorithms, Alfred Aho, John Hopcroft, and Jeffery Ullman, Addison–Wesley, 1983.

BCMP — Open, Closed, and Mixed Networks of Queues with Different Classes of Customers, F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, *Journal of the ACM*, 22(2), 248-260

Chartrand — Introductory Graph Theory, Gary Chartrand, Dover Publications, 1985.

CIP — CEOS Catalogue Interoperability Protocol (CIP) Specification – Release A, Version 2.2, March 1997.

CTN — Interoperable Catalogues System (ICS) Collection Technical Note (CTN), ECS 170-WP-006-003, Version 1.0, July 1996.

Feller — An Introduction to Probability Theory and Its Applications, Third Edition, William Feller, John Wiley & Sons, 1968.

Hartsfield and Ringel — Pearls in Graph Theory, Nora Hartsfield and Gerhard Ringel, Academic Press, 1990.

SDD — Interoperable Catalogue System (ICS) System Design Document (SDD), Version 1.2, March, 1997.

Tarjan — Data Structures and Network Algorithms, Robert Endre Tarjan, Society for Industrial and Applied Mathematics, 1983.

URD — CEOS Interoperable Catalogues System (ICS) User Requirement Document (URD), Version 2.2, March 1997.

Appendix B. Delphic Study

During the June 1997 meeting of the PTT Core Team, a Delphic study was held to determine what the team felt were the most likely values of certain parameters. The parameters of concern were the average number of collections per collection (A), the average collection depth (D), and the average inventory (catalogue) query service time.

A Delphic study attempts to use a group’s expertise. It does this by having the members of the group make independent estimates of specified parameters, without consulting one another. The estimates are tabulated and then averaged. In the case of the PTT, members were asked to give an estimate a low value such that 90% of all ICS systems would have a value greater than that for the parameter. They also were asked to estimate a high value such that only 10% of the ICS system would have a value greater than that. Finally, they were asked to estimate what they felt were would be the average. The estimates and the averages are listed in the table below.

One member of the PTT was not present for the initial portion of the study. His estimates of the averages were made afterwards. They are listed in table below, but not calculated into the average values.

Table B-1. Results of PTT Core Team Delphic Study

Name	**A**			**D**			**d_{inv}**		
	90 %	**Ave**	**10 %**	**90 %**	**Ave**	**10 %**	**90 %**	**Ave**	**10 %**
George Percivall	2	5	30	1	5	10	2	30	600
Brian McCloud	2	7	50	1	3	6	0.5	30	600
Steve Smith	2	3.5	30	1.5	2	5	20	90	1200
Richard Gobel	1	5	20	3	5	10	1	30	1800
Ladson Hayes	1	5	10	1	2	5	5	50	86,400
Clive Best	1.5	3	9	2	4	8	3	20	900
Lou Reich		20			2		3	30	600
AVERAGE	1.6	4.75	24.8	1.6	3.5	7.3	4.9	40	15,100

This page intentionally left blank.

Abbreviations and Acronyms

BCMP	A type of queuing network, see reference BCMP.
CEOS	Committee on Earth Observation Satellites
CDB	Collection Database
CIP	Catalogue Interoperability Protocol
CTN	Collection Technical Note
DAG	Directed Acyclic Graph
ECS	EOSDIS Core System
FCFS	First Come, First Served
HMI	Human–Machine Interface
ICS	Interoperable Catalogue System
IID	Independent and Identically Distributed
LAN	Local Area Network
LCFS–PR	Last Come, First Served with Preemptive Resume
M/M/m	Kendall notation for a type of queue.
NASA	National Aeronautics and Space Administration (US)
NSP	Network Service Provider(s)
PS	Processor Sharing
PTT	Protocol Task Team
RM	Retrieval Manager
SDD	System Design Document
TBD	To Be Determined
TDM	Time Division Multiplexed
URD	User Requirement Document
WAN	Wide Area Network
WWW	World Wide Web

This page intentionally left blank.

Glossary of Terms

The following technical terms, as defined below, are used in this document:

Directed Tree
The directed version of a tree (i.e., a digraph formed from a tree by replacing the tree's edges with arcs).

Directed Graph
A directed graph consists of a set of nodes and a set of arcs connecting pairs of nodes. An arc is directed starting at the tail and terminating at the head. An arc may be traversed only in the direction from the tail to the head. More mathematically, it is a finite nonempty set *V* together with an irreflexive relation *R* on *V*. The interested reader may wish to consult Chartrand.

Graph
A graph is a finite nonempty set *V* together with an irreflexive, symmetric relation *R* on *V*. The interested reader may wish to consult Chartrand.

Response Time
The response time is the time from the point at which a user (either human or machine) requests a service until the time when a response is presented to that user. It includes time spent in queues, transmission delays, and the like. In the context of ICS and this study, the requesting user is typically is a CIP origin.

Rooted Tree
A directed tree that has precisely one vertex (called the root) that has an indegree of zero (0), that is, has no parent. This definition is from Hartsfield and Ringel. Tarjan uses a stricter definition that uses the more general "tree" instead of "directed tree". He then goes on to state "when appropriate we shall regard the edges of a rooted tree as directed."

Service Time
The service time is the time required to actually perform a service. It is the elapsed process time once the service is granted. It does not include any queuing time.

Tree
A tree is a connected graph that contains no cycles (Reference Chartrand).

Uniform Tree
A uniform tree is either a simple tree or a directed tree with a uniform number of items per node and each branch of the root node extending to the same depth.

This page intentionally left blank.